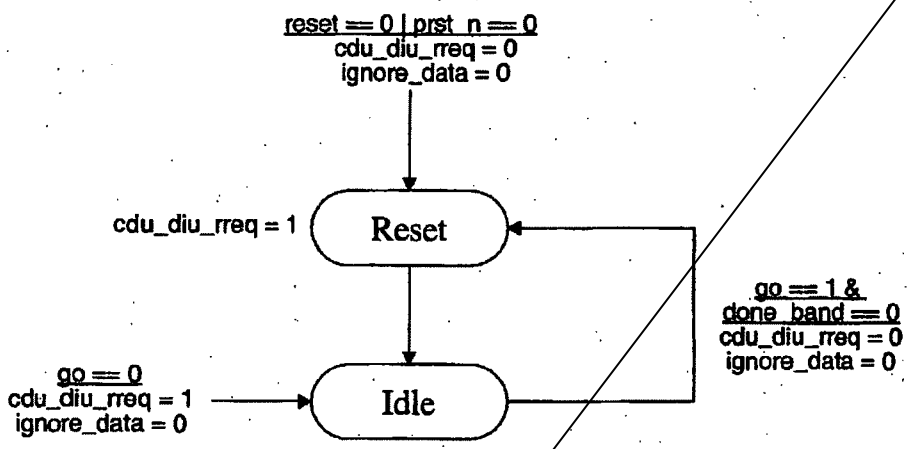

reset == 0 | prst_n == 0
cdu_diu_rreq = 0
ignore_data = 0
cdu_diu_rreq = 1
Reset
go == 1 &
done_band == 0
cdu_diu_rreq = 0
ignore_data = 0
go == 0
cdu_diu_rreq = 1
ignore_data = 0
Idle

FIG. 1

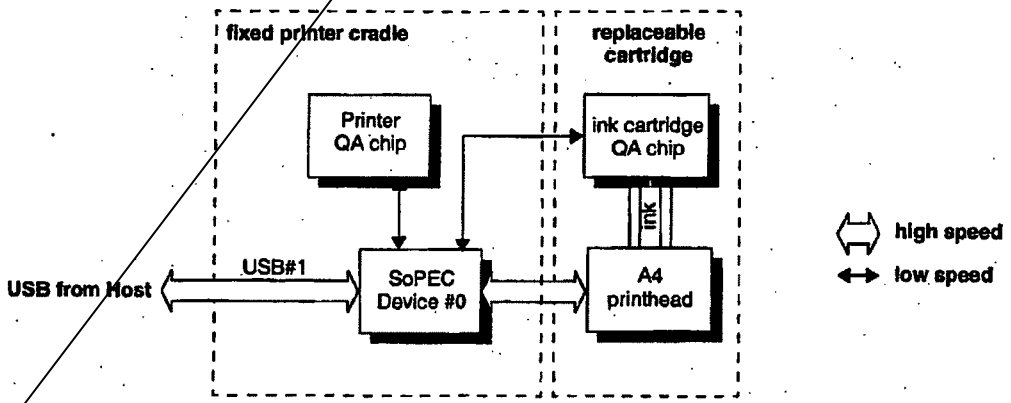

fixed printer cradle
replaceable cartridge
Printer QA chip
ink cartridge QA chip
ink
USB from Host
USB#1
SoPEC Device #0
A4 printhead
high speed
low speed

FIG. 2

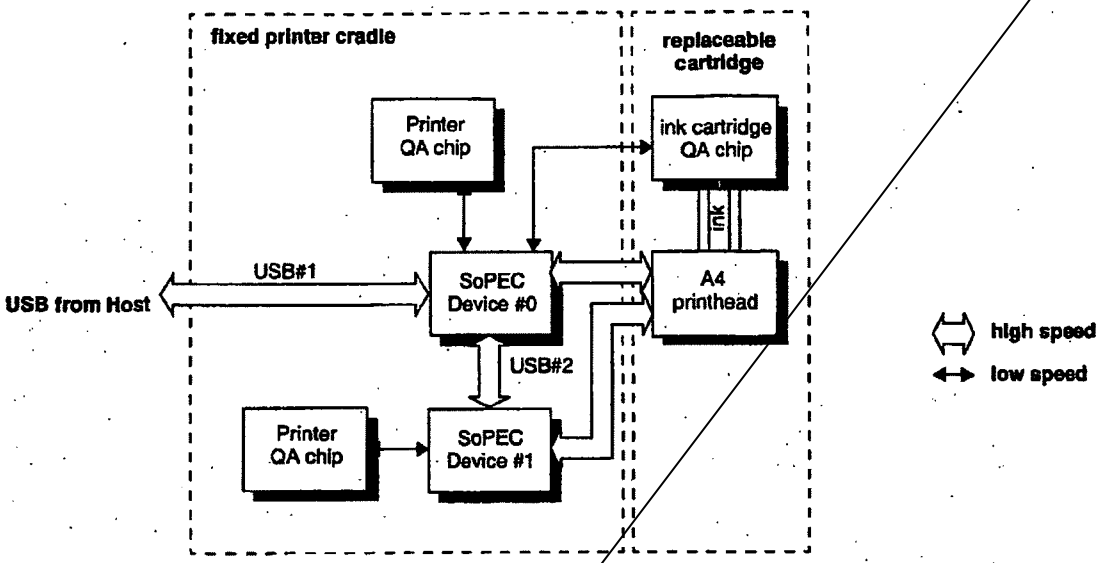

fixed printer cradle
replaceable cartridge
Printer QA chip
ink cartridge QA chip
ink
USB#1
USB from Host
SoPEC Device #0
A4 printhead
USB#2
Printer QA chip
SoPEC Device #1
high speed
low speed

FIG. 3

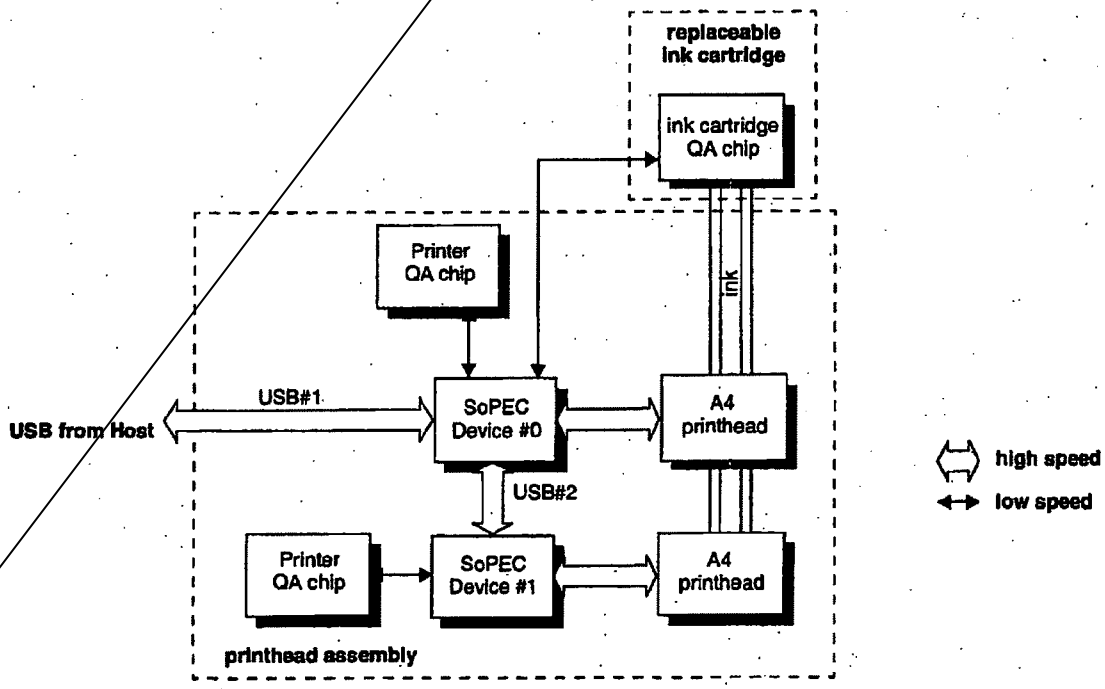

replaceable ink cartridge
ink cartridge QA chip
Printer QA chip
ink
USB#1
USB from Host
SoPEC Device #0
A4 printhead
USB#2
Printer QA chip
SoPEC Device #1
A4 printhead
printhead assembly
high speed
low speed

FIG. 4

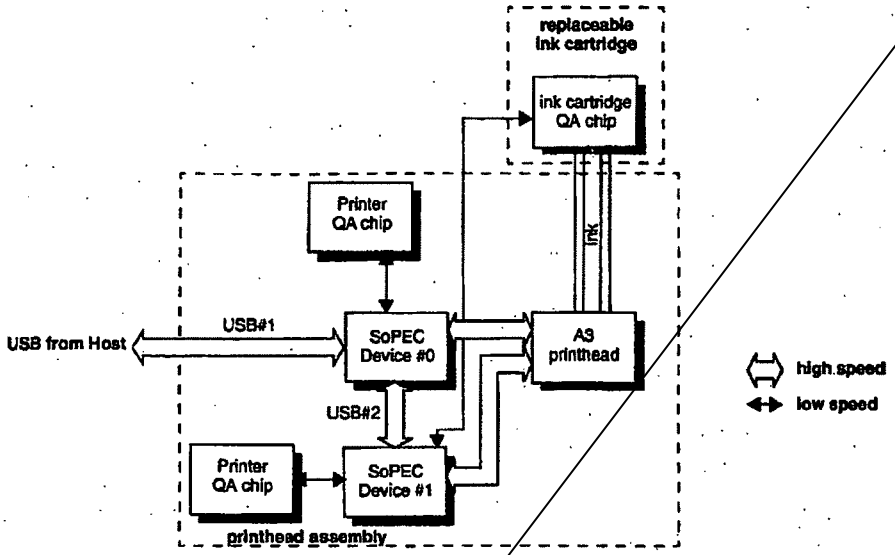

replaceable ink cartridge
ink cartridge QA chip
Printer QA chip
ink
USB from Host
USB#1
SoPEC Device #0
A3 printhead
high speed
low speed
USB#2
Printer QA chip
SoPEC Device #1
printhead assembly

FIG. 5

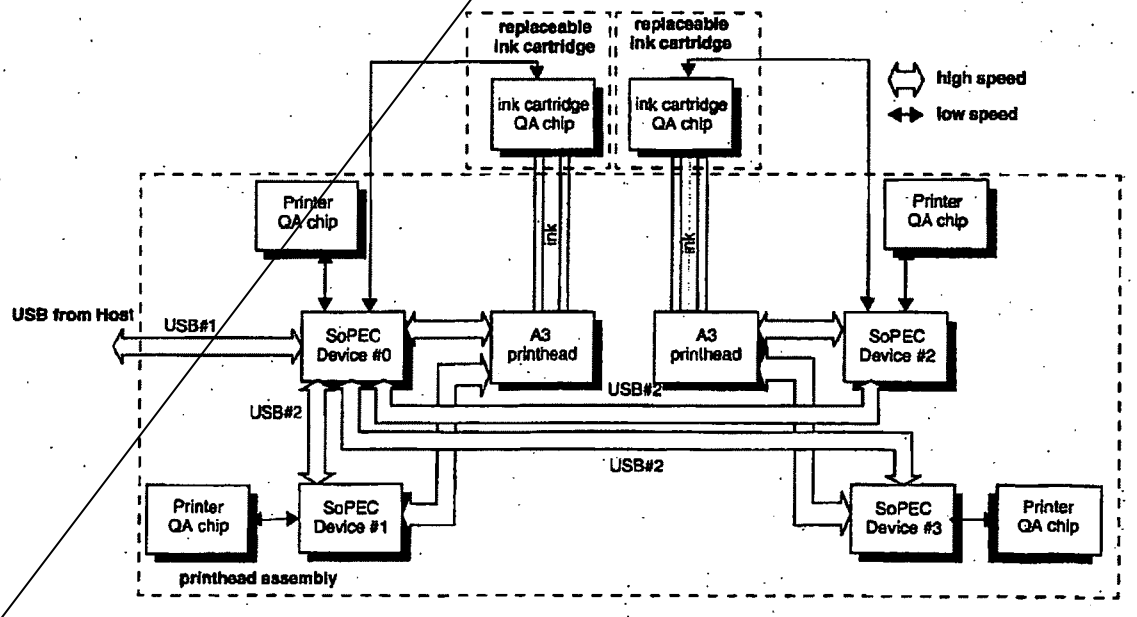

replaceable ink cartridge
replaceable ink cartridge
high speed
low speed
ink cartridge QA chip
ink cartridge QA chip
Printer QA chip
Printer QA chip
ink
ink
USB from Host
USB#1
SoPEC Device #0
A3 printhead
A3 printhead
SoPEC Device #2
USB#2
USB#2
USB#2
Printer QA chip
SoPEC Device #1
SoPEC Device #3
Printer QA chip
printhead assembly

FIG. 6

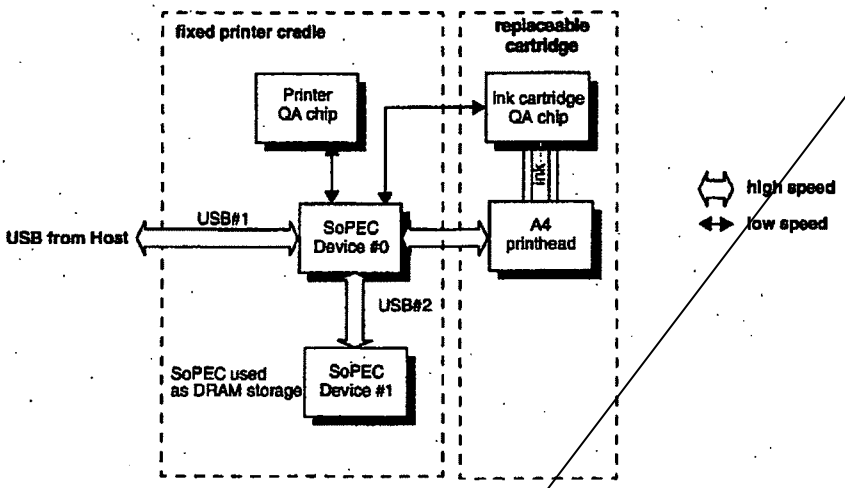

fixed printer cradle
replaceable cartridge
Printer QA chip
ink cartridge QA chip
ink
USB from Host
USB#1
SoPEC Device #0
A4 printhead
USB#2
SoPEC used as DRAM storage
SoPEC Device #1
high speed
low speed

FIG. 7

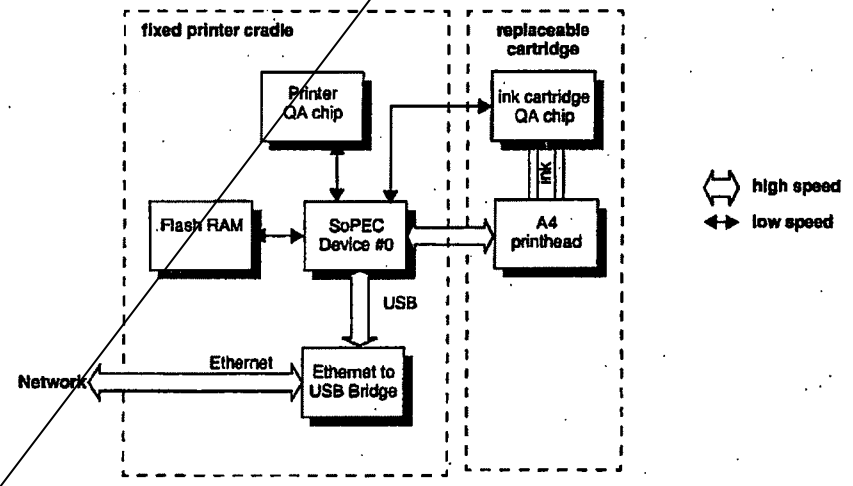

fixed printer cradle
replaceable cartridge
Printer QA chip
ink cartridge QA chip
ink
Flash RAM
SoPEC Device #0
A4 printhead
USB
Network
Ethernet
Ethernet to USB Bridge
high speed
low speed

FIG. 8

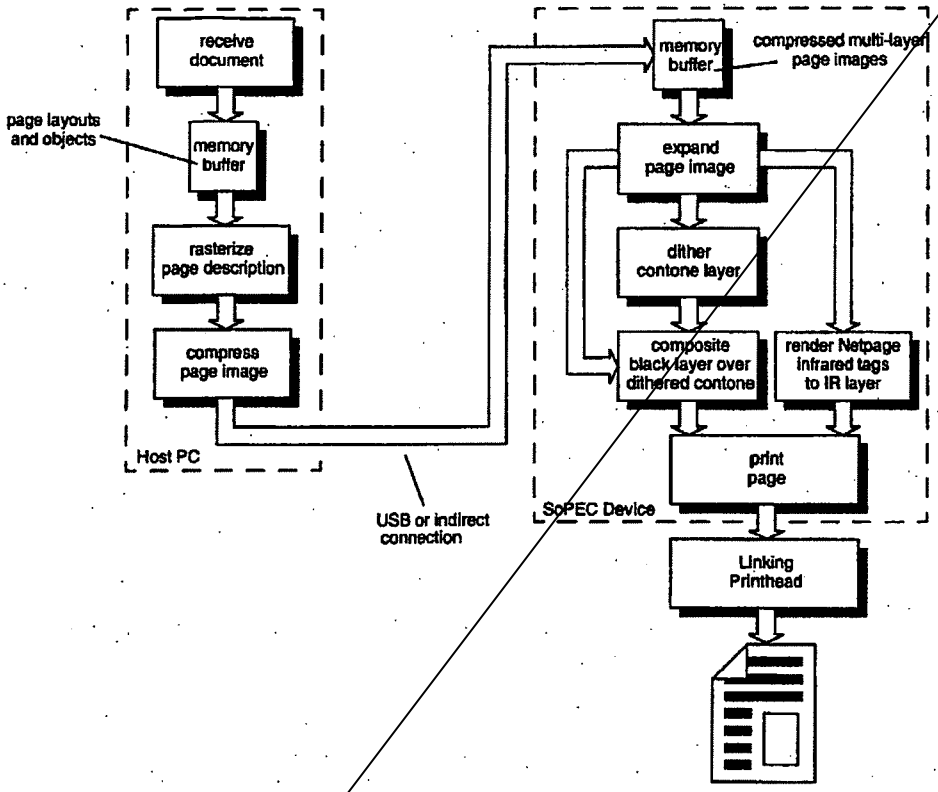

receive document
page layouts and objects
memory buffer
rasterize page description
compress page image
Host PC
USB or indirect connection
memory buffer
compressed multi-layer page images
expand page image
dither contone layer
composite black layer over dithered contone
render Netpage infrared tags to IR layer
print page
SoPEC Device
Linking Printhead

FIG. 9

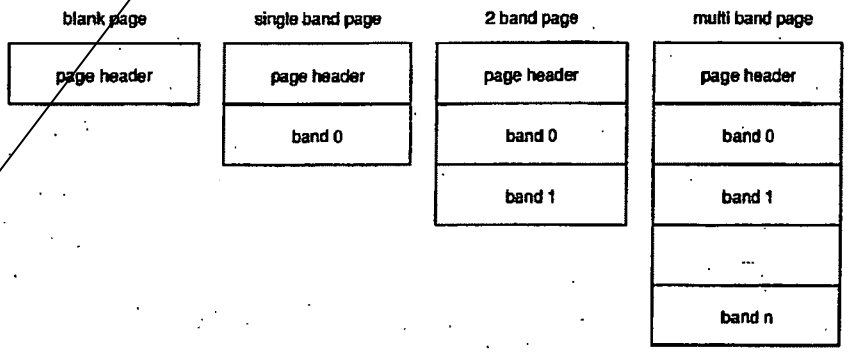

blank page
page header
single band page
page header
band 0
2 band page
page header
band 0
band 1
multi band page
page header
band 0
band 1
...
band n

FIG. 10

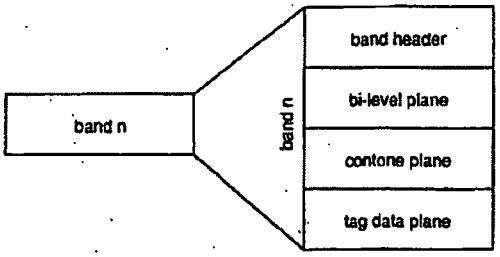


FIG. 11

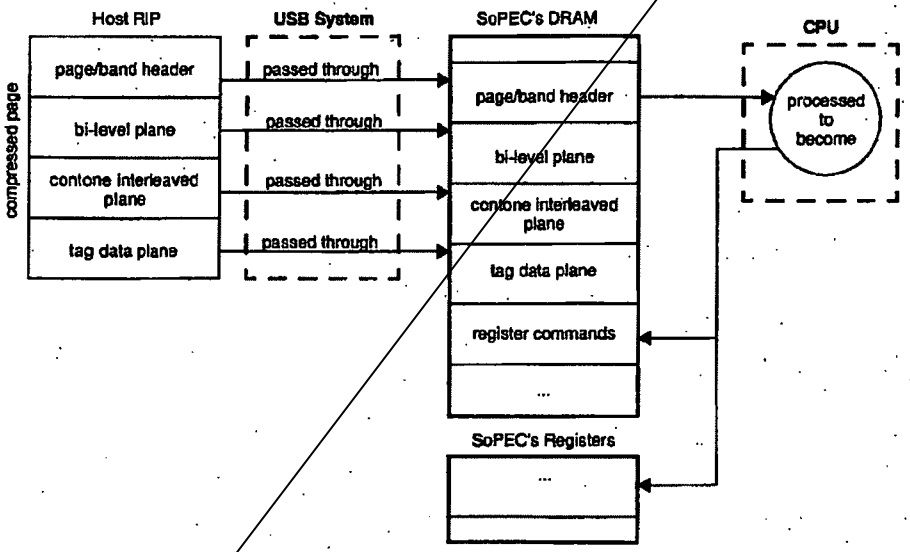


FIG. 12

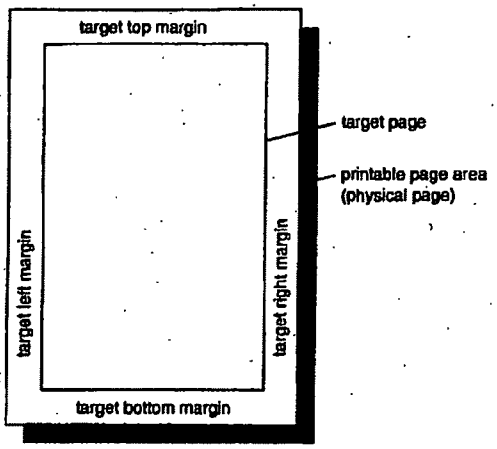


FIG. 13

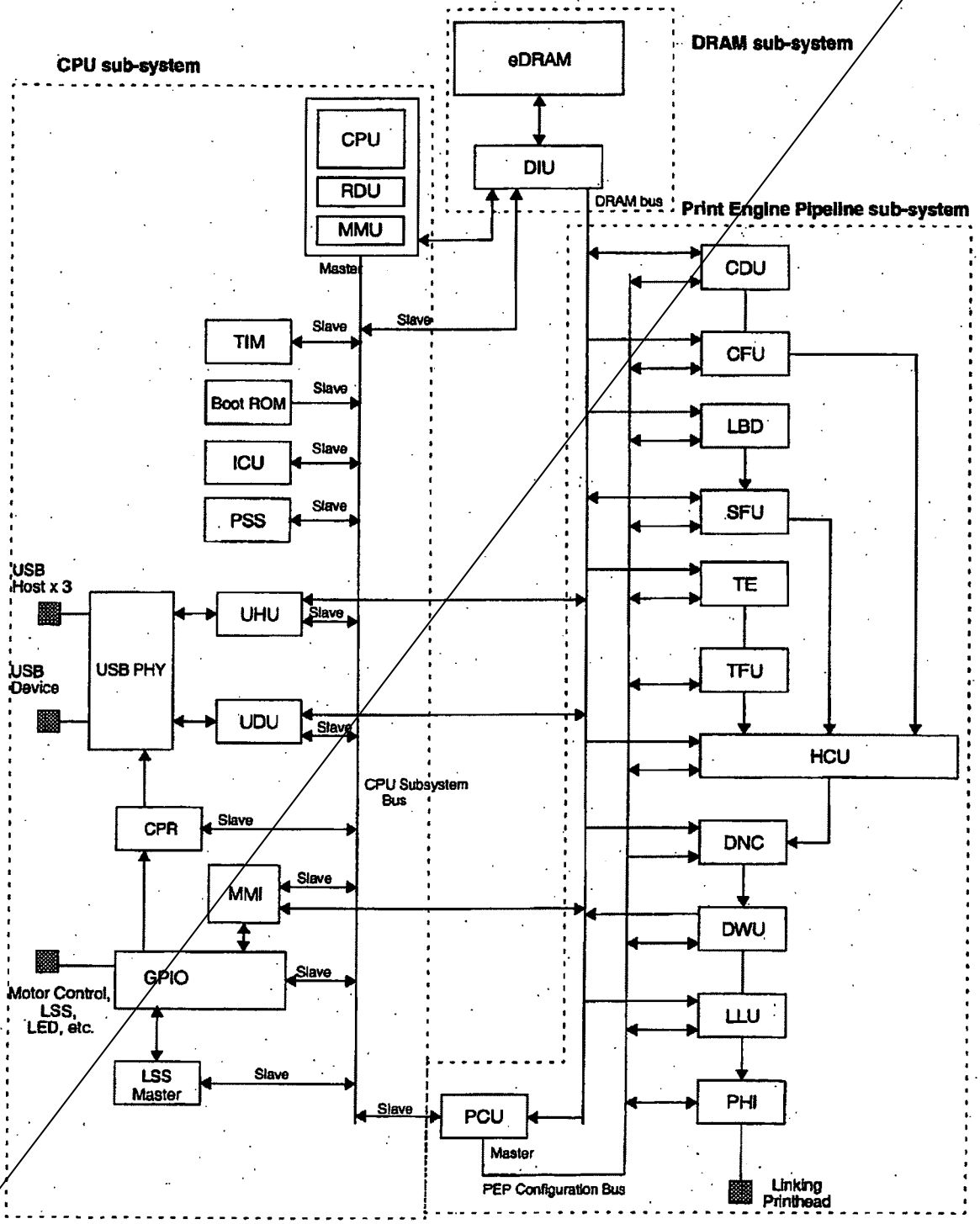

CPU sub-system
DRAM sub-system
eDRAM
CPU
RDU
MMU
DIU
DRAM bus
Print Engine Pipeline sub-system
Master
Slave
TIM
Boot ROM
ICU
PSS
CDU
CFU
LBD
SFU
TE
TFU
HCU
DNC
DWU
LLU
PHI
USB Host x 3
USB Device
USB PHY
UHU
UDU
CPU Subsystem Bus
CPR
MMI
GPIO
Motor Control, LSS, LED, etc.
LSS Master
PCU
PEP Configuration Bus
Linking Printhead

FIG. 14

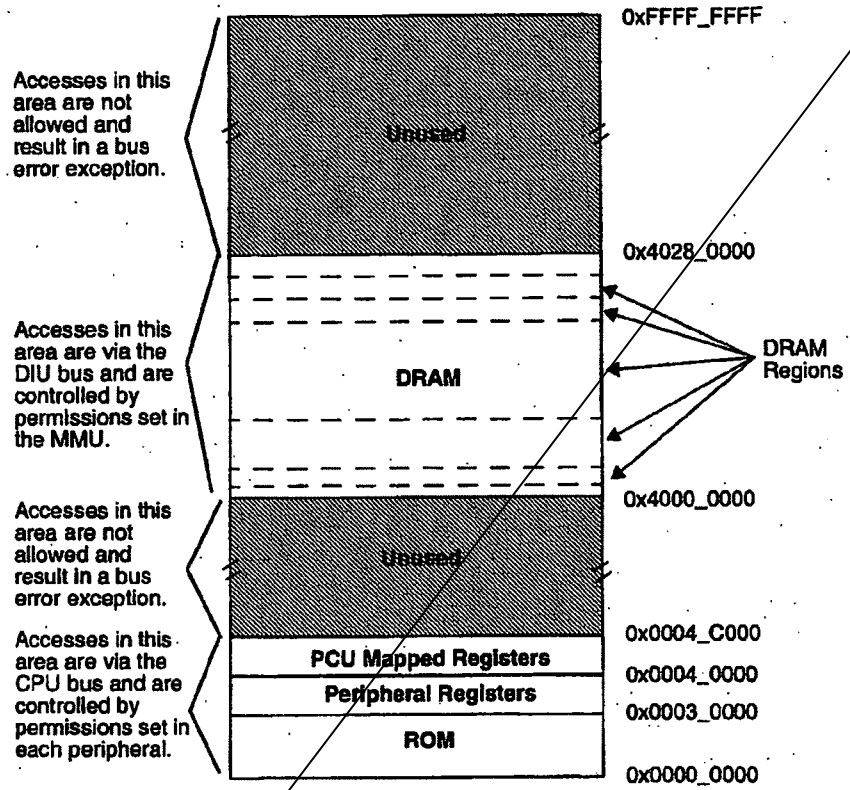
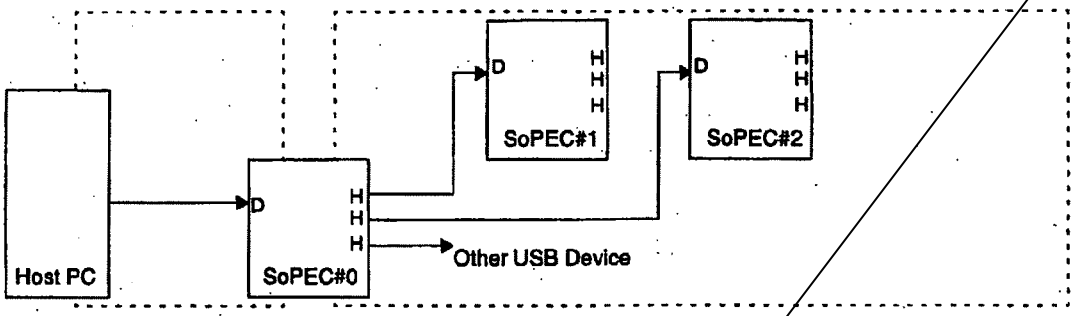
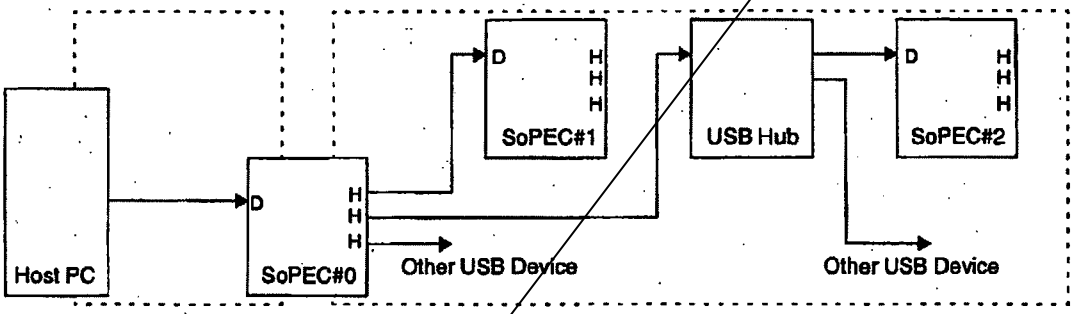

0xFFFF_FFFF
Accesses in this area are not allowed and result in a bus error exception.
Unused
0x4028_0000
Accesses in this area are via the DIU bus and are controlled by permissions set in the MMU.
DRAM
DRAM Regions
0x4000_0000
Accesses in this area are not allowed and result in a bus error exception.
Unused
Accesses in this area are via the CPU bus and are controlled by permissions set in each peripheral.
0x0004_C000
PCU Mapped Registers
0x0004_0000
Peripheral Registers
0x0003_0000
ROM
0x0000_0000

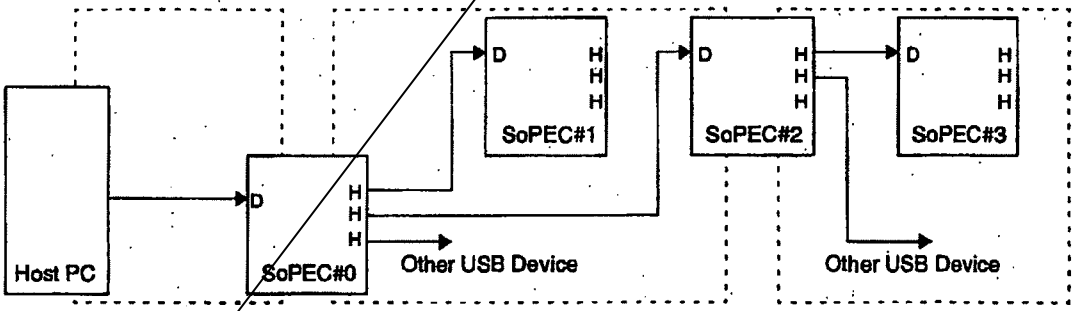
FIG. 15



Case 1: One Printer USB Bus with no Hub chips, up to 3 Devices on the bus



Case 2: One Printer USB Bus including Hub chip, more than 3 Devices on the bus



Case 3: Two Printer USB Busses, up to 3 devices on each bus

FIG. 16

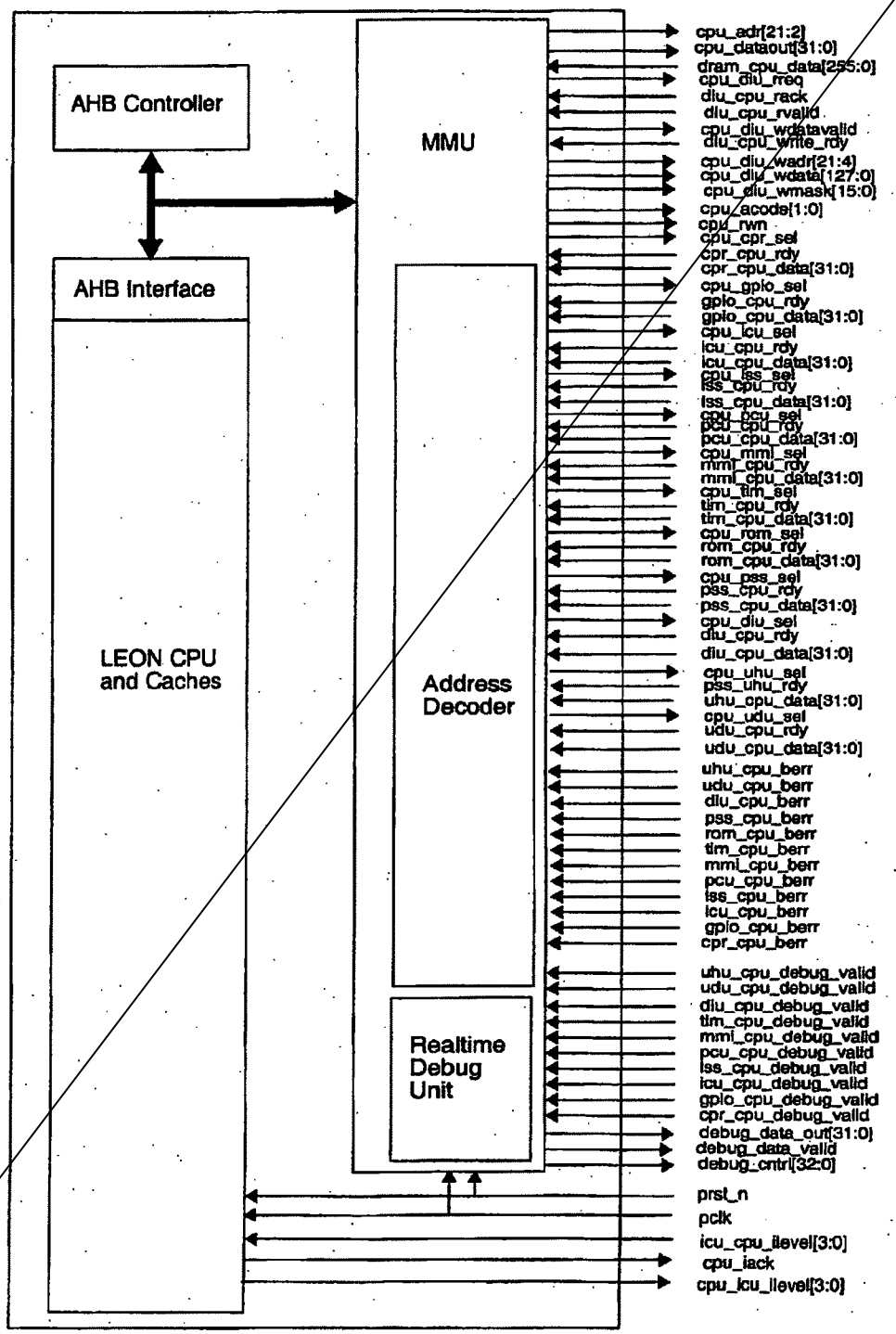

AHB Controller
AHB Interface
LEON CPU and Caches
MMU
Address Decoder
Realtime Debug Unit
cpu_adr[21:2]
cpu_dataout[31:0]
dram_cpu_data[255:0]
cpu_diu_req
diu_cpu_rack
diu_cpu_rvalid
cpu_diu_wdatavalid
diu_cpu_write_rdy
cpu_diu_wadr[21:4]
cpu_diu_wdata[127:0]
cpu_diu_wmask[15:0]
cpu_acode[1:0]
cpu_rwn
cpu_cpr_sel
cpr_cpu_rdy
cpr_cpu_data[31:0]
cpu_gpio_sel
gpio_cpu_rdy
gpio_cpu_data[31:0]
cpu_icu_sel
icu_cpu_rdy
icu_cpu_data[31:0]
cpu_iss_sel
iss_cpu_rdy
iss_cpu_data[31:0]
cpu_pcu_sel
pcu_cpu_rdy
pcu_cpu_data[31:0]
cpu_mmi_sel
mmi_cpu_rdy
mmi_cpu_data[31:0]
cpu_tim_sel
tim_cpu_rdy
tim_cpu_data[31:0]
cpu_rom_sel
rom_cpu_rdy
rom_cpu_data[31:0]
cpu_pss_sel
pss_cpu_rdy
pss_cpu_data[31:0]
cpu_diu_sel
diu_cpu_rdy
diu_cpu_data[31:0]
cpu_uhu_sel
pss_uhu_rdy
uhu_cpu_data[31:0]
cpu_udu_sel
udu_cpu_rdy
udu_cpu_data[31:0]
uhu_cpu_berr
udu_cpu_berr
diu_cpu_berr
pss_cpu_berr
rom_cpu_berr
tim_cpu_berr
mmi_cpu_berr
pcu_cpu_berr
iss_cpu_berr
icu_cpu_berr
gpio_cpu_berr
cpr_cpu_berr
uhu_cpu_debug_valid
udu_cpu_debug_valid
diu_cpu_debug_valid
tim_cpu_debug_valid
mmi_cpu_debug_valid
pcu_cpu_debug_valid
iss_cpu_debug_valid
icu_cpu_debug_valid
gpio_cpu_debug_valid
cpr_cpu_debug_valid
debug_data_out[31:0]
debug_data_valid
debug_cntrl[32:0]
prst_n
pclk
icu_cpu_ilevel[3:0]
cpu_iack
cpu_icu_ilevel[3:0]

FIG. 17

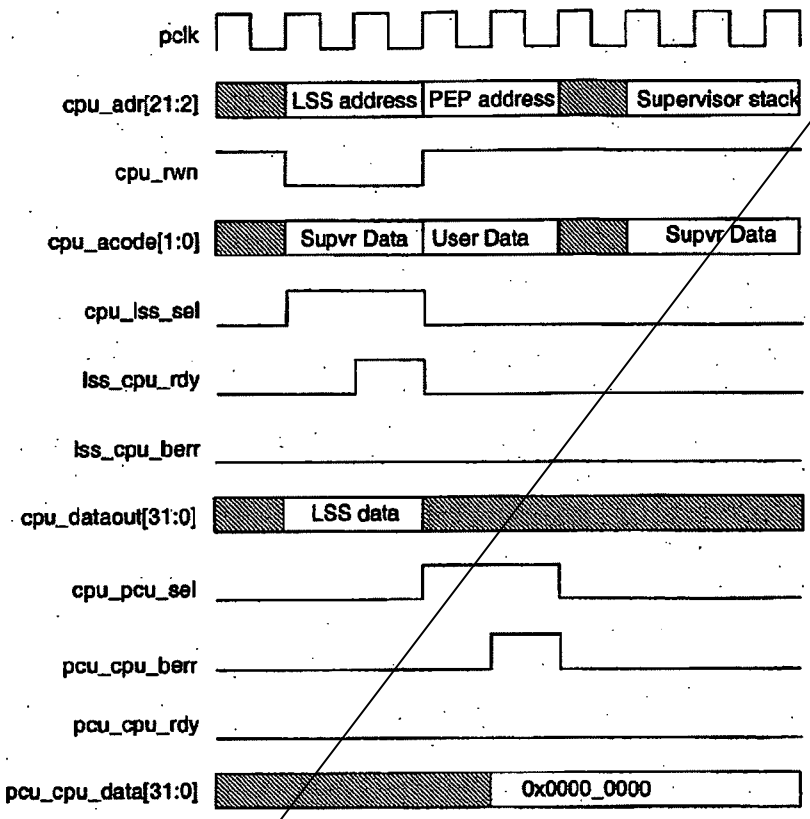

pclk
cpu_adr[21:2]
LSS address
PEP address
Supervisor stack
cpu_rwn
cpu_acode[1:0]
Supvr Data
User Data
Supvr Data
cpu_lss_sel
lss_cpu_rdy
lss_cpu_berr
cpu_dataout[31:0]
LSS data
cpu_pcu_sel
pcu_cpu_berr
pcu_cpu_rdy
pcu_cpu_data[31:0]
0x0000_0000

FIG. 18

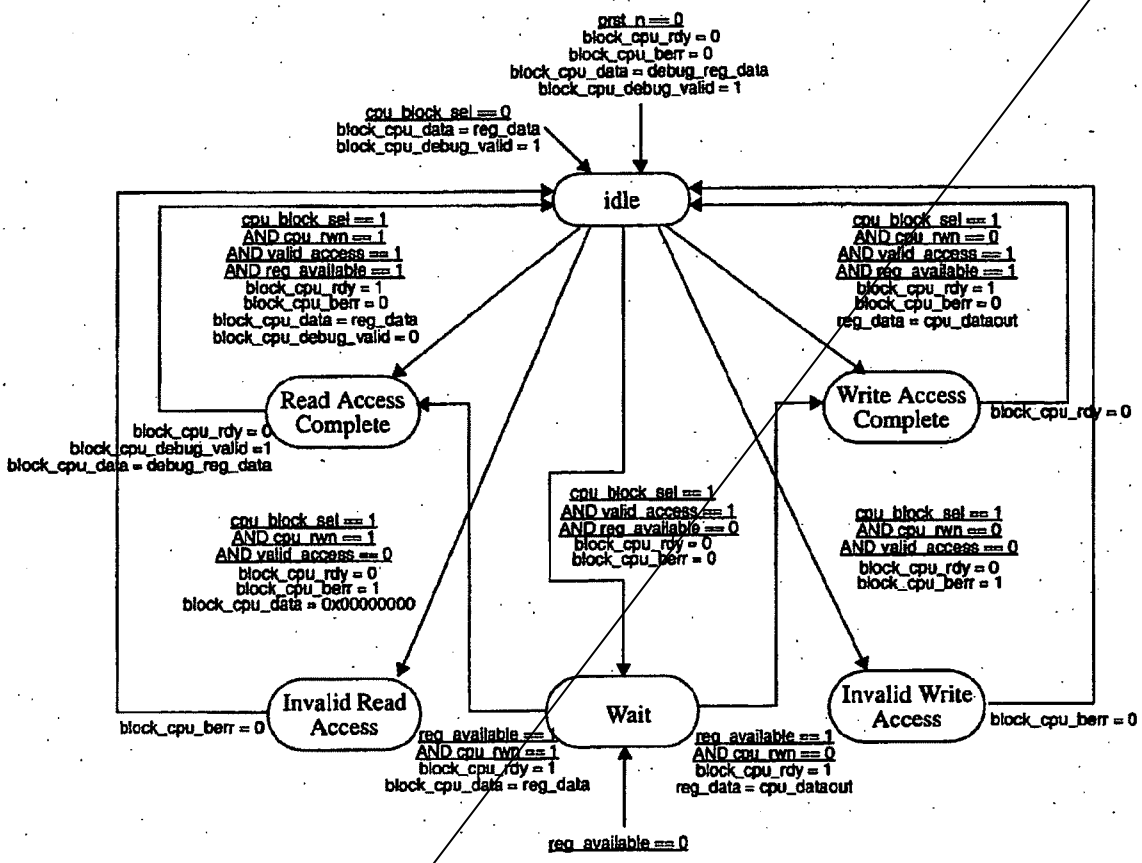

prst_n == 0
block_cpu_rdy = 0
block_cpu_berr = 0
block_cpu_data = debug_reg_data
block_cpu_debug_valid = 1
cpu_block_sel == 0
block_cpu_data = reg_data
block_cpu_debug_valid = 1
idle
cpu_block_sel == 1
AND cpu_rwn == 1
AND valid_access == 1
AND reg_available == 1
block_cpu_rdy = 1
block_cpu_berr = 0
block_cpu_data = reg_data
block_cpu_debug_valid = 0
cpu_block_sel == 1
AND cpu_rwn == 0
AND valid_access == 1
AND reg_available == 1
block_cpu_rdy = 1
block_cpu_berr = 0
reg_data = cpu_dataout
Read Access Complete
block_cpu_rdy = 0
block_cpu_debug_valid =1
block_cpu_data = debug_reg_data
Write Access Complete
block_cpu_rdy = 0
cpu_block_sel == 1
AND cpu_rwn == 1
AND valid_access == 0
block_cpu_rdy = 0
block_cpu_berr = 1
block_cpu_data = 0x00000000
cpu_block_sel == 1
AND valid_access == 1
AND reg_available == 0
block_cpu_rdy = 0
block_cpu_berr = 0
cpu_block_sel == 1
AND cpu_rwn == 0
AND valid_access == 0
block_cpu_rdy = 0
block_cpu_berr = 1
Invalid Read Access
block_cpu_berr = 0
Wait
Invalid Write Access
block_cpu_berr = 0
reg_available == 1
AND cpu_rwn == 1
block_cpu_rdy = 1
block_cpu_data = reg_data
reg_available == 1
AND cpu_rwn == 0
block_cpu_rdy = 1
reg_data = cpu_dataout
reg_available == 0

FIG. 19

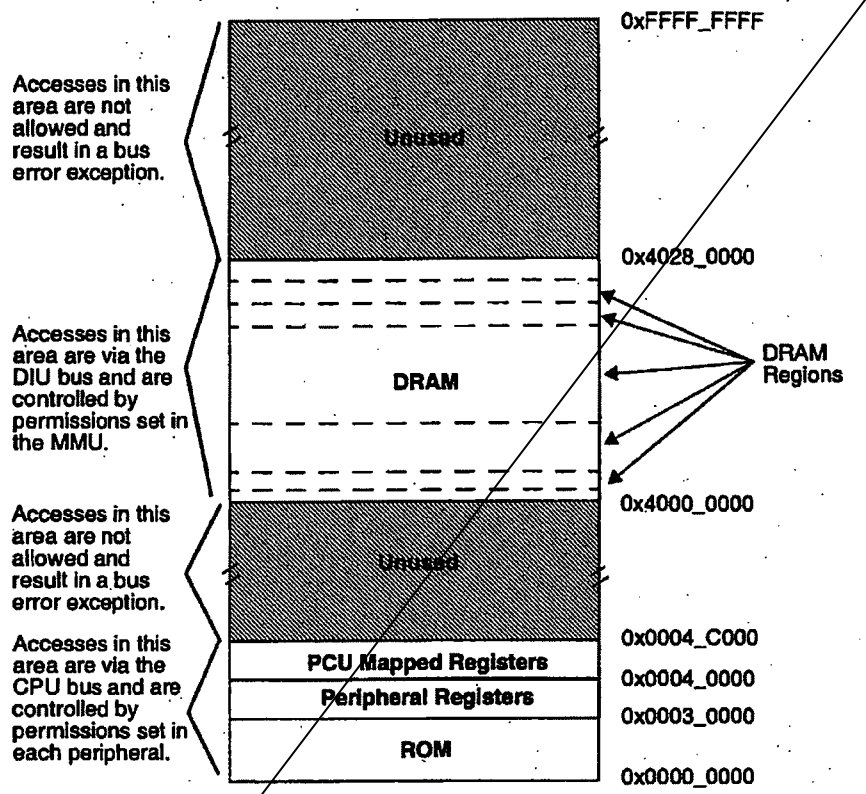

Accesses in this area are not allowed and result in a bus error exception.
Accesses in this area are via the DIU bus and are controlled by permissions set in the MMU.
Accesses in this area are not allowed and result in a bus error exception.
Accesses in this area are via the CPU bus and are controlled by permissions set in each peripheral.
Unused
DRAM
Unused
PCU Mapped Registers
Peripheral Registers
ROM
0xFFFF_FFFF
0x4028_0000
0x4000_0000
0x0004_C000
0x0004_0000
0x0003_0000
0x0000_0000
DRAM Regions

FIG. 20

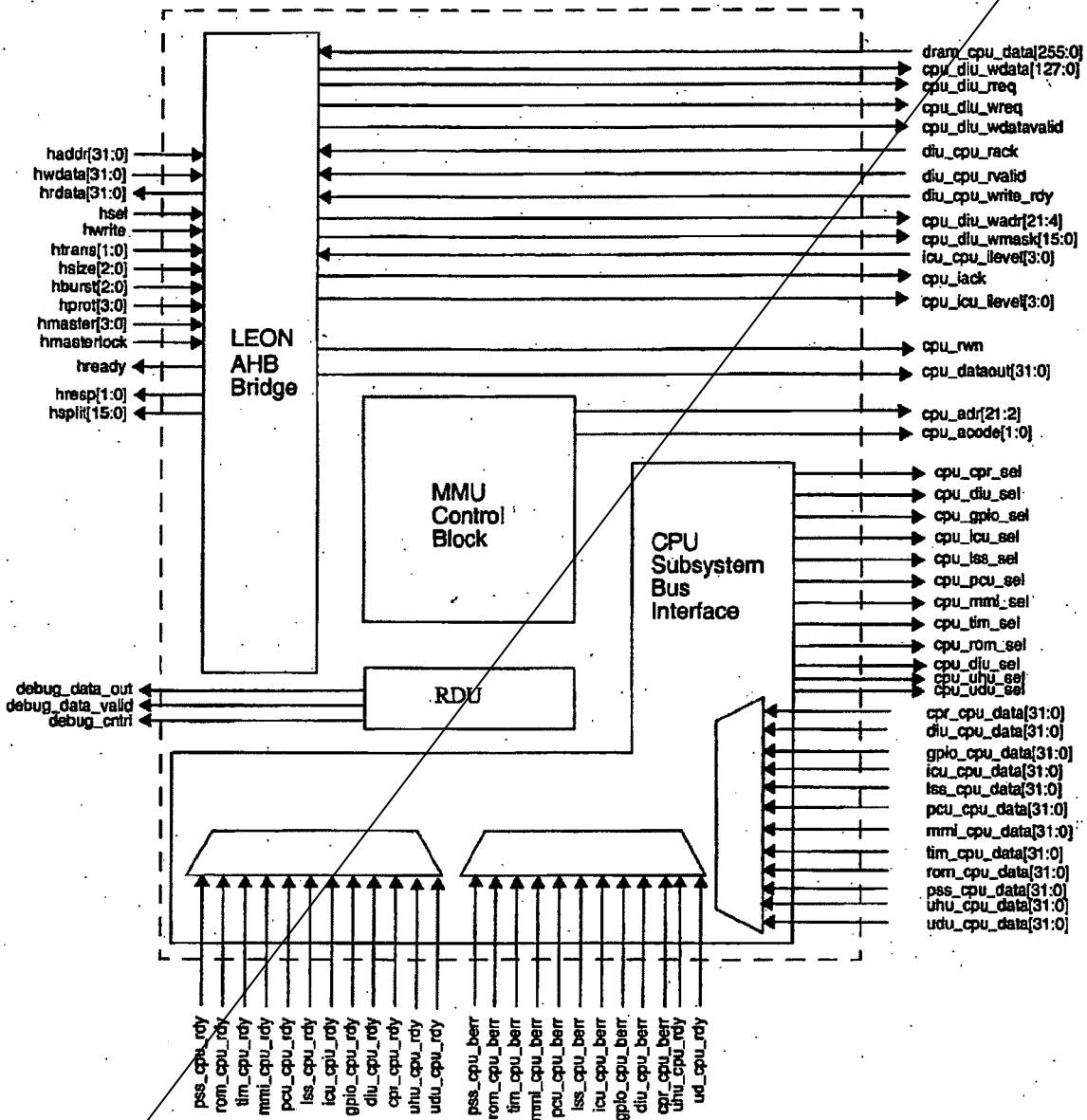

haddr[31:0]
hwdata[31:0]
hrdata[31:0]
hsel
hwrite
htrans[1:0]
hsize[2:0]
hburst[2:0]
hprot[3:0]
hmaster[3:0]
hmasterlock
hready
hresp[1:0]
hsplit[15:0]
LEON AHB Bridge
MMU Control Block
RDU
CPU Subsystem Bus Interface
debug_data_out
debug_data_valid
debug_cntrl
dram_cpu_data[255:0]
cpu_diu_wdata[127:0]
cpu_diu_req
cpu_diu_wreq
cpu_diu_wdatavalid
diu_cpu_rack
diu_cpu_rvalid
diu_cpu_write_rdy
cpu_diu_wadr[21:4]
cpu_diu_wmask[15:0]
icu_cpu_ilevel[3:0]
cpu_iack
cpu_icu_ilevel[3:0]
cpu_rwn
cpu_dataout[31:0]
cpu_adr[21:2]
cpu_acode[1:0]
cpu_cpr_sel
cpu_diu_sel
cpu_gpio_sel
cpu_icu_sel
cpu_iss_sel
cpu_pcu_sel
cpu_mmi_sel
cpu_tim_sel
cpu_rom_sel
cpu_diu_sel
cpu_uhu_sel
cpu_udu_sel
cpr_cpu_data[31:0]
diu_cpu_data[31:0]
gpio_cpu_data[31:0]
icu_cpu_data[31:0]
iss_cpu_data[31:0]
pcu_cpu_data[31:0]
mmi_cpu_data[31:0]
tim_cpu_data[31:0]
rom_cpu_data[31:0]
pss_cpu_data[31:0]
uhu_cpu_data[31:0]
udu_cpu_data[31:0]
pss_cpu_rdy
rom_cpu_rdy
tim_cpu_rdy
mmi_cpu_rdy
pcu_cpu_rdy
iss_cpu_rdy
icu_cpu_rdy
gpio_cpu_rdy
diu_cpu_rdy
cpr_cpu_rdy
uhu_cpu_rdy
udu_cpu_rdy
pss_cpu_berr
rom_cpu_berr
tim_cpu_berr
mmi_cpu_berr
pcu_cpu_berr
iss_cpu_berr
icu_cpu_berr
gpio_cpu_berr
diu_cpu_berr
cpr_cpu_berr
uhu_cpu_rdy
ud_cpu_rdy

FIG. 21

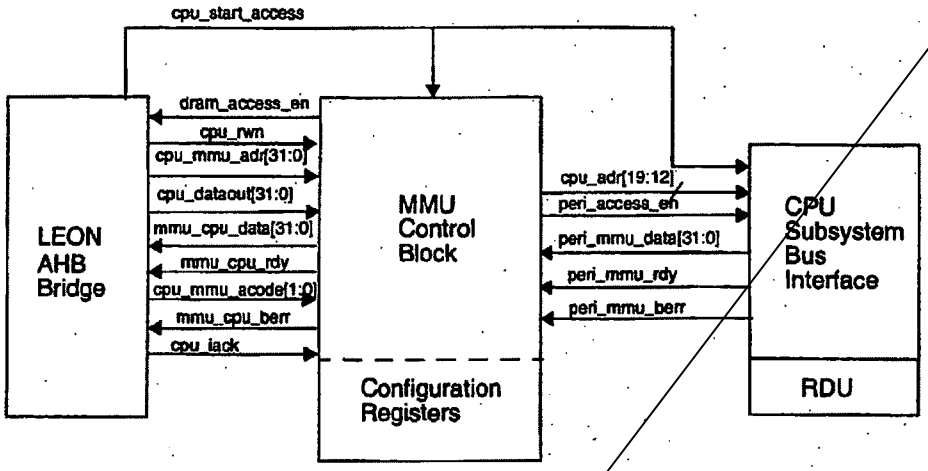

cpu_start_access
LEON
AHB
Bridge
dram_access_en
cpu_rwn
cpu_mmu_adr[31:0]
cpu_dataout[31:0]
mmu_cpu_data[31:0]
mmu_cpu_rdy
cpu_mmu_acode[1:0]
mmu_cpu_berr
cpu_iack
MMU
Control
Block
Configuration
Registers
cpu_adr[19:12]
peri_access_en
peri_mmu_data[31:0]
peri_mmu_rdy
peri_mmu_berr
CPU
Subsystem
Bus
Interface
RDU

FIG. 22

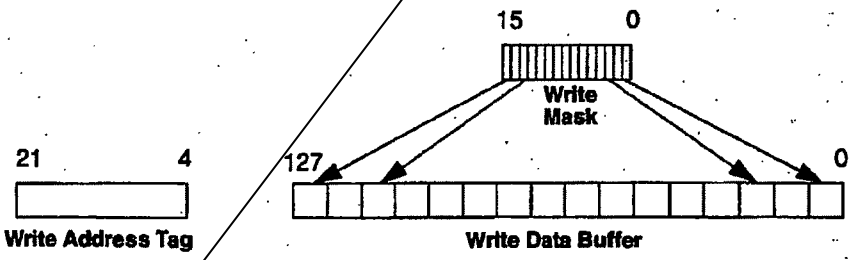

15
0
Write
Mask
21
4
127
0
Write Address Tag
Write Data Buffer

FIG. 23

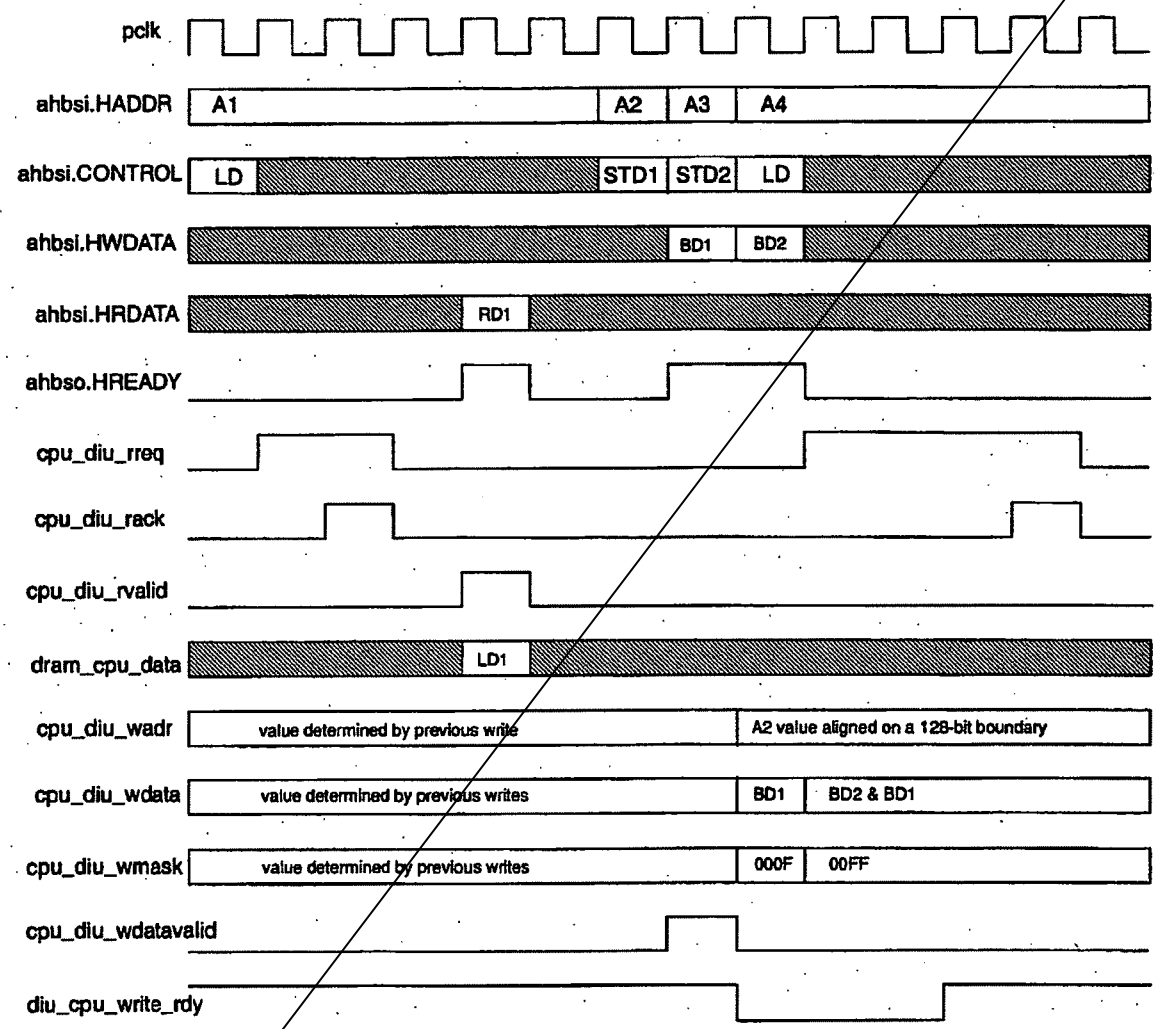

pclk
ahbsi.HADDR
A1
A2
A3
A4
ahbsi.CONTROL
LD
STD1
STD2
LD
ahbsi.HWDATA
BD1
BD2
ahbsi.HRDATA
RD1
ahbso.HREADY
cpu_diu_rreq
cpu_diu_rack
cpu_diu_rvalid
dram_cpu_data
LD1
cpu_diu_wadr
value determined by previous write
A2 value aligned on a 128-bit boundary
cpu_diu_wdata
value determined by previous writes
BD1
BD2 & BD1
cpu_diu_wmask
value determined by previous writes
000F
00FF
cpu_diu_wdatavalid
diu_cpu_write_rdy

FIG. 24

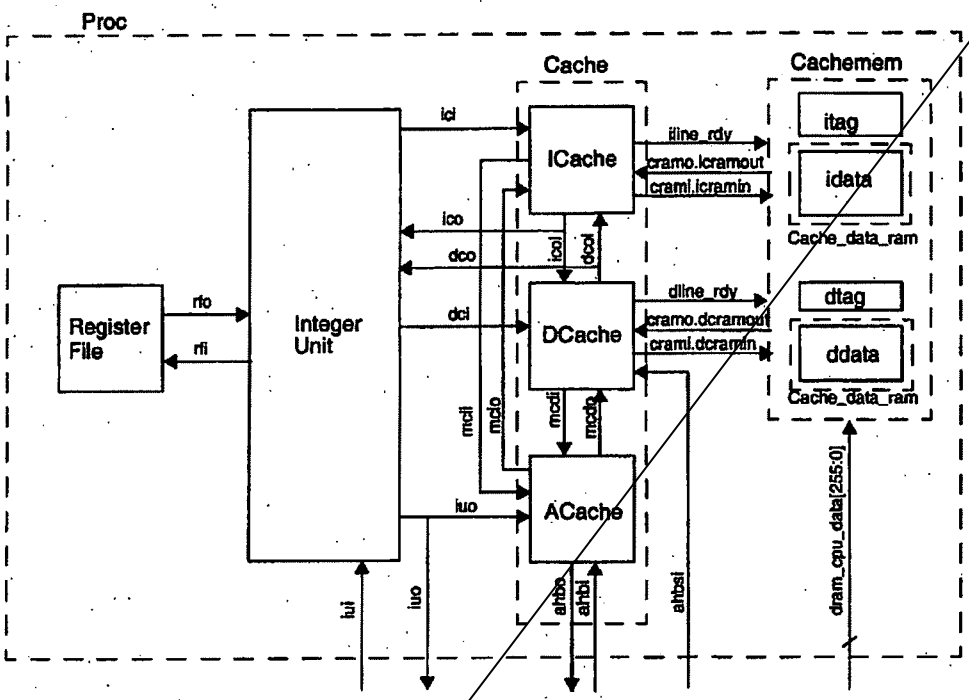

Proc
Cache
Cachemem
Register File
rfo
rfi
Integer Unit
ici
ico
dco
dci
iuo
iui
iuo
ICache
DCache
ACache
icol
dcol
mcti
mcto
mcdi
mcdo
ahbso
ahbsi
ahbsi
iline_rdy
cramo.icramout
crami.icramin
dline_rdy
cramo.dcramout
crami.dcramin
itag
idata
Cache_data_ram
dtag
ddata
Cache_data_ram
dram_cpu_data[255:0]

FIG. 25

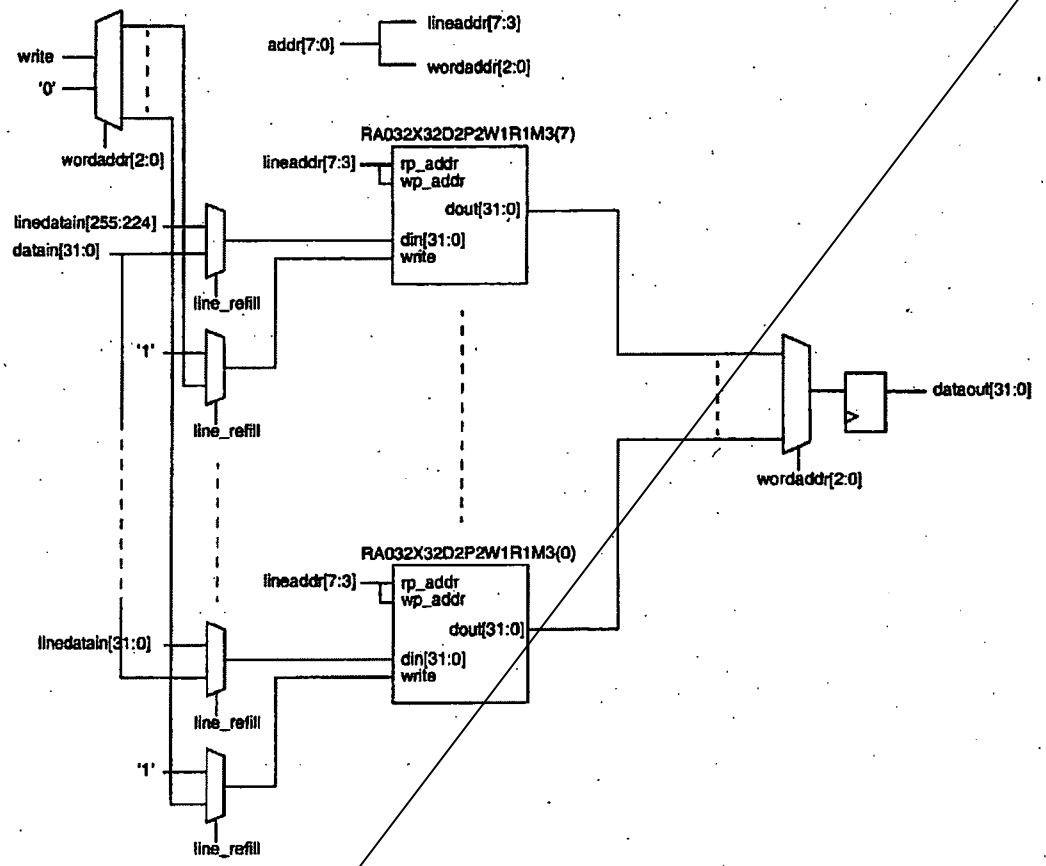

write
'0'
wordaddr[2:0]
addr[7:0]
lineaddr[7:3]
wordaddr[2:0]
RA032X32D2P2W1R1M3(7)
lineaddr[7:3]
rp_addr
wp_addr
dout[31:0]
din[31:0]
write
linedatain[255:224]
datain[31:0]
line_refill
'1'
line_refill
dataout[31:0]
wordaddr[2:0]
RA032X32D2P2W1R1M3(0)
lineaddr[7:3]
rp_addr
wp_addr
dout[31:0]
din[31:0]
write
linedatain[31:0]
line_refill
'1'
line_refill

FIG. 26

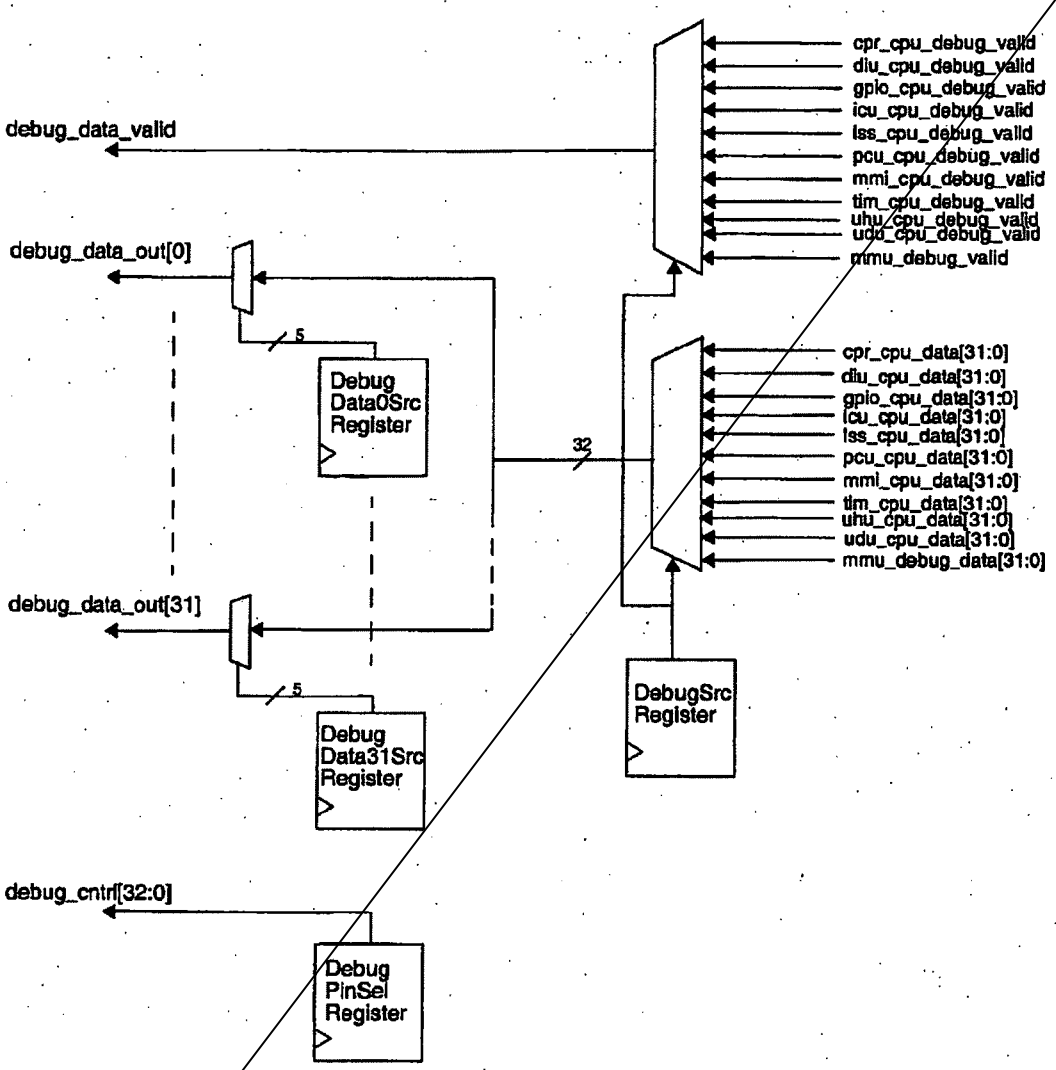

cpr_cpu_debug_valid
diu_cpu_debug_valid
gpio_cpu_debug_valid
icu_cpu_debug_valid
lss_cpu_debug_valid
pcu_cpu_debug_valid
mmi_cpu_debug_valid
tim_cpu_debug_valid
uhu_cpu_debug_valid
udu_cpu_debug_valid
mmu_debug_valid
debug_data_valid
debug_data_out[0]
5
Debug Data0Src Register
32
cpr_cpu_data[31:0]
diu_cpu_data[31:0]
gpio_cpu_data[31:0]
icu_cpu_data[31:0]
lss_cpu_data[31:0]
pcu_cpu_data[31:0]
mmi_cpu_data[31:0]
tim_cpu_data[31:0]
uhu_cpu_data[31:0]
udu_cpu_data[31:0]
mmu_debug_data[31:0]
debug_data_out[31]
5
Debug Data31Src Register
DebugSrc Register
debug_cntrl[32:0]
Debug PinSel Register

FIG. 27

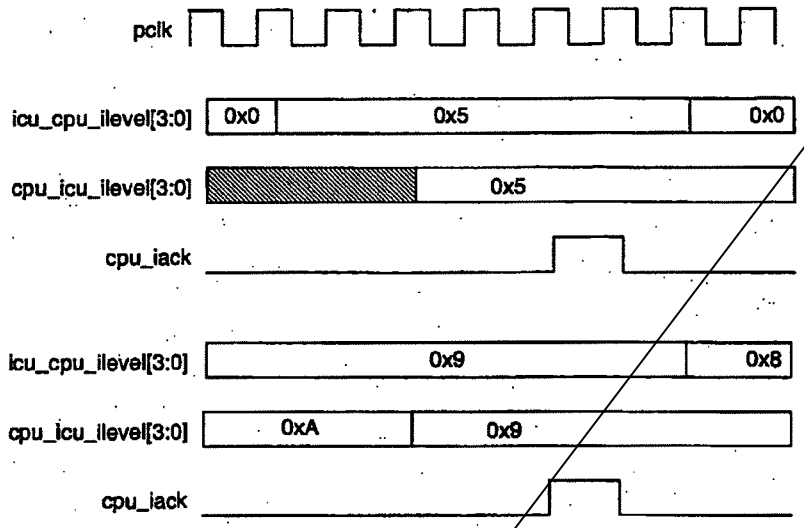

pclk
icu_cpu_ilevel[3:0]
0x0
0x5
0x0
cpu_icu_ilevel[3:0]
0x5
cpu_iack
icu_cpu_ilevel[3:0]
0x9
0x8
cpu_icu_ilevel[3:0]
0xA
0x9
cpu_iack

FIG. 28

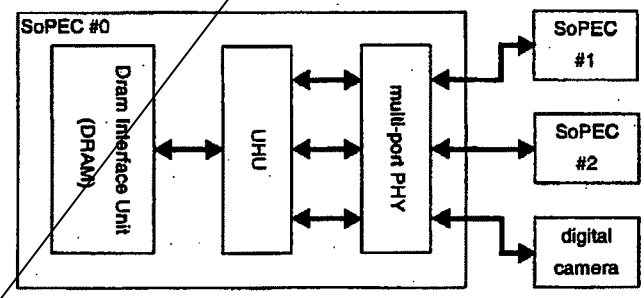

SoPEC #0
Dram Interface Unit (DRAM)
UHU
multi-port PHY
SoPEC #1
SoPEC #2
digital camera

FIG. 29

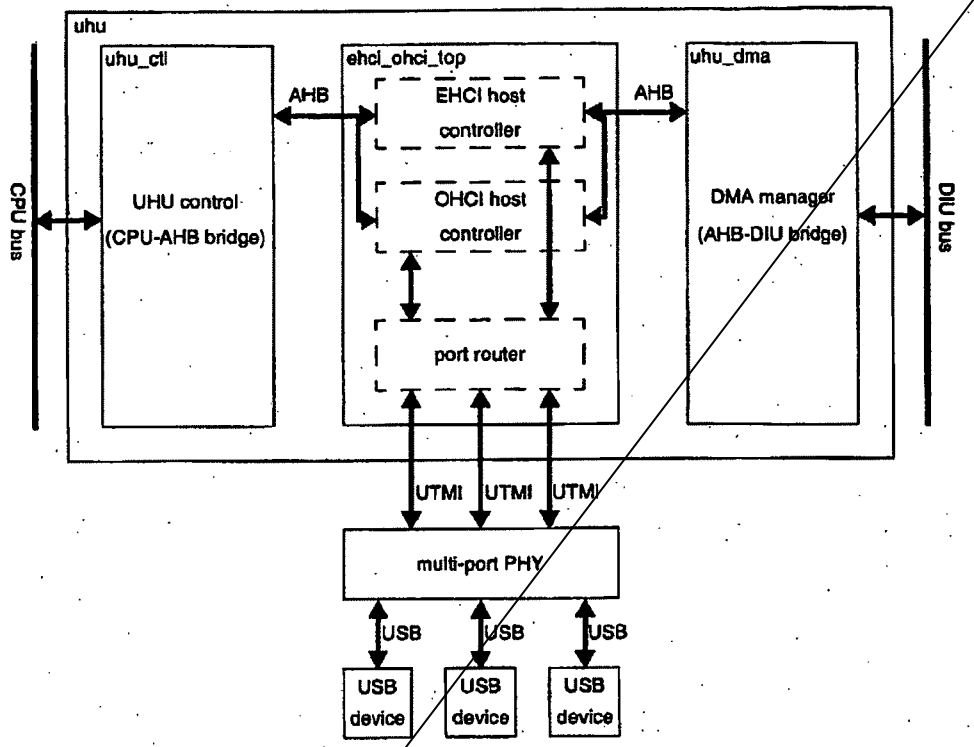

uhu
uhu_ctl
UHU control
(CPU-AHB bridge)
ehci_ohci_top
EHCI host
controller
OHCI host
controller
port router
uhu_dma
DMA manager
(AHB-DIU bridge)
AHB
AHB
CPU bus
DIU bus
UTMI
UTMI
UTMI
multi-port PHY
USB
USB
USB
USB
device
USB
device
USB
device

FIG. 30

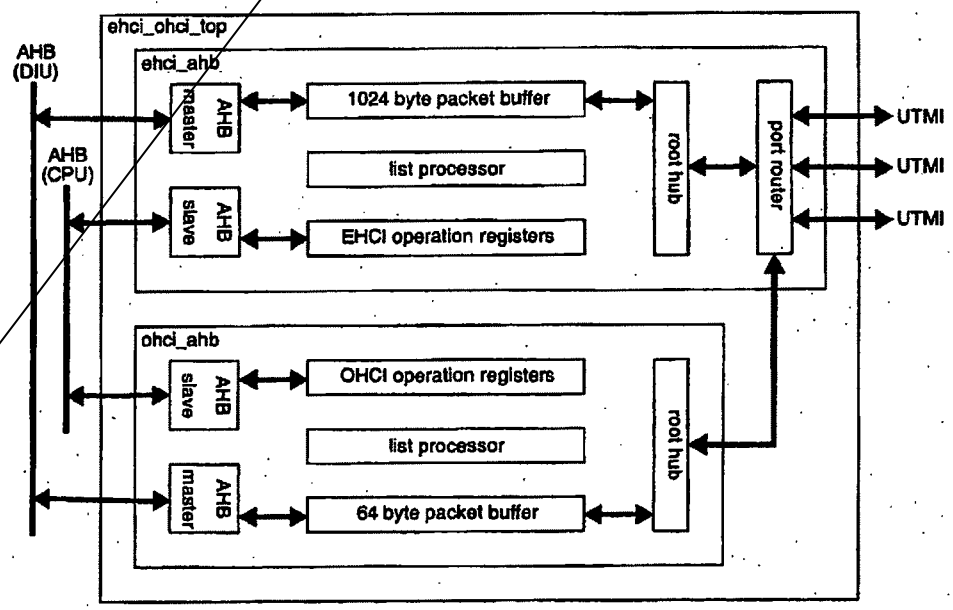

ehci_ohci_top
AHB
(DIU)
AHB
(CPU)
ehci_ahb
master
AHB
slave
AHB
1024 byte packet buffer
list processor
EHCI operation registers
root hub
port router
UTMI
UTMI
UTMI
ohci_ahb
slave
AHB
master
AHB
OHCI operation registers
list processor
64 byte packet buffer
root hub

FIG. 31

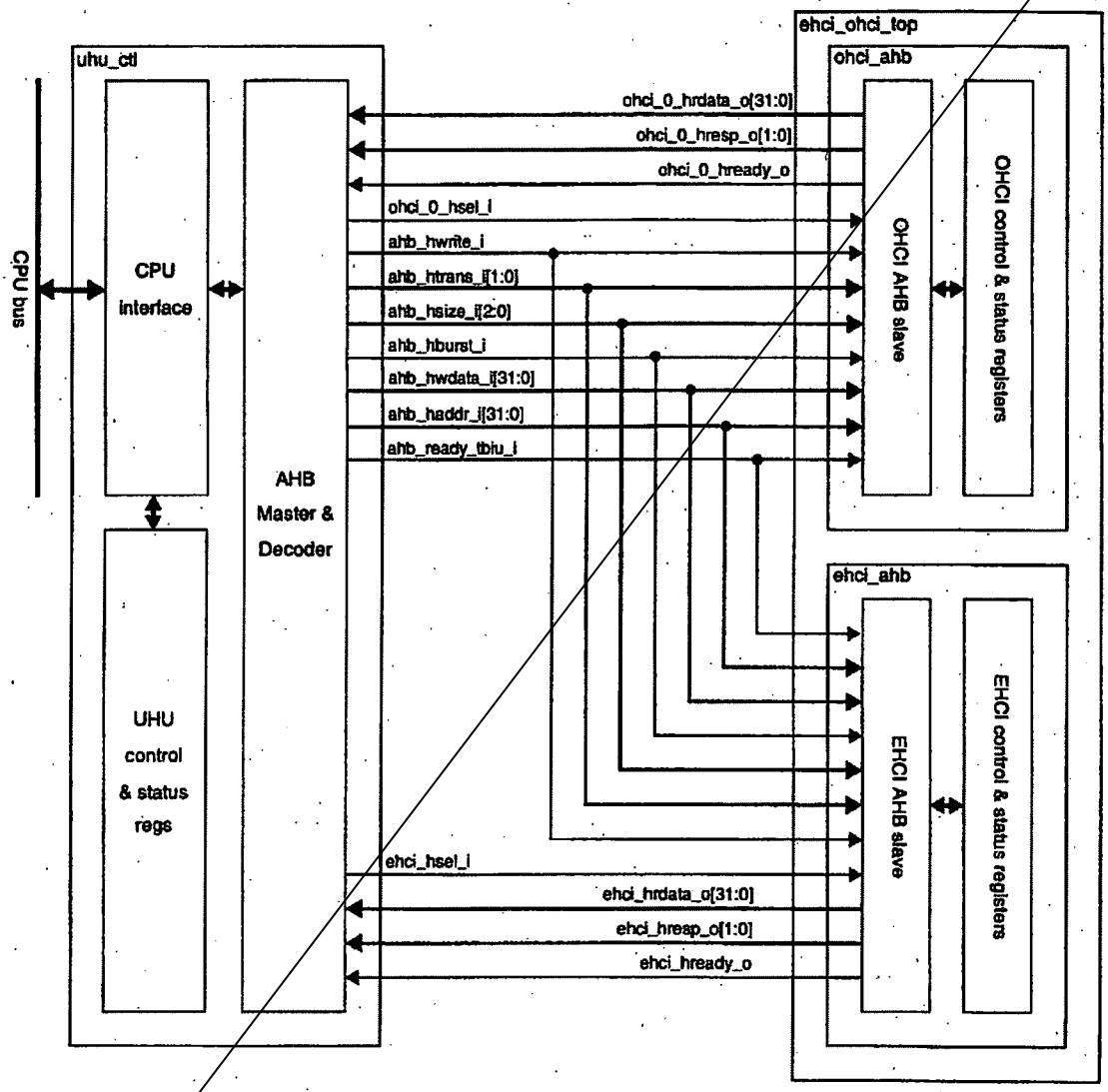

ehci_ohci_top
uhu_ctl
ohci_ahb
ohci_0_hrdata_o[31:0]
ohci_0_hresp_o[1:0]
ohci_0_hready_o
ohci_0_hsel_i
ahb_hwrite_i
ahb_htrans_i[1:0]
ahb_hsize_i[2:0]
ahb_hburst_i
ahb_hwdata_i[31:0]
ahb_haddr_i[31:0]
ahb_ready_tbiu_i
CPU bus
CPU interface
AHB Master & Decoder
OHCI AHB slave
OHCI control & status registers
ehci_ahb
UHU control & status regs
EHCI AHB slave
EHCI control & status registers
ehci_hsel_i
ehci_hrdata_o[31:0]
ehci_hresp_o[1:0]
ehci_hready_o

FIG. 32

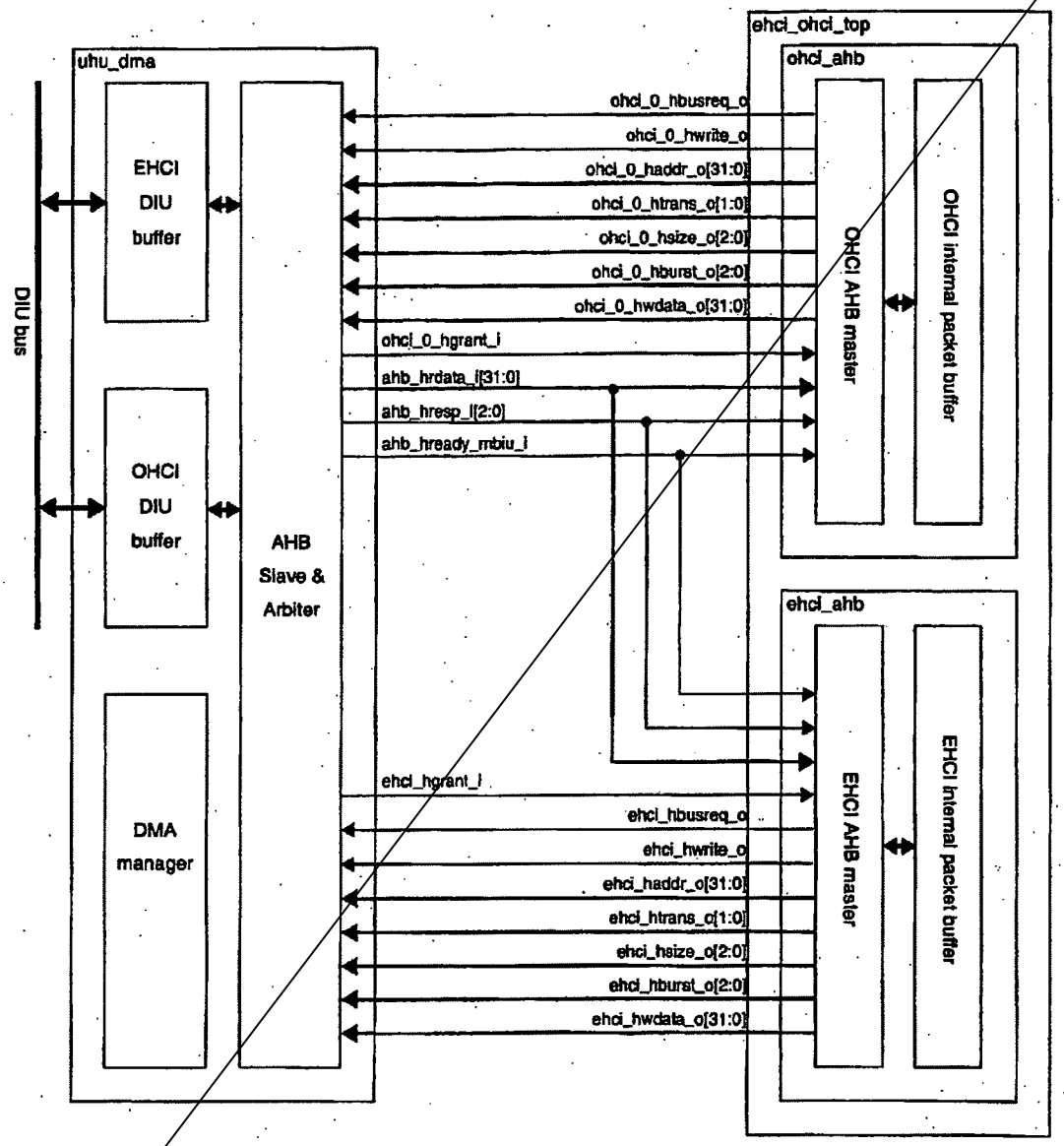

ehci_ohci_top
uhu_dma
ohci_ahb
DIU bus
EHCI DIU buffer
OHCI DIU buffer
DMA manager
AHB Slave & Arbiter
ohci_0_hbusreq_o
ohci_0_hwrite_o
ohci_0_haddr_o[31:0]
ohci_0_htrans_o[1:0]
ohci_0_hsize_o[2:0]
ohci_0_hburst_o[2:0]
ohci_0_hwdata_o[31:0]
ohci_0_hgrant_i
ahb_hrdata_i[31:0]
ahb_hresp_i[2:0]
ahb_hready_mbiu_i
OHCI AHB master
OHCI internal packet buffer
ehci_ahb
ehci_hgrant_i
ehci_hbusreq_o
ehci_hwrite_o
ehci_haddr_o[31:0]
ehci_htrans_o[1:0]
ehci_hsize_o[2:0]
ehci_hburst_o[2:0]
ehci_hwdata_o[31:0]
EHCI AHB master
EHCI internal packet buffer

FIG. 33

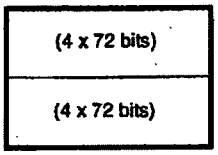

(4 x 72 bits)
(4 x 72 bits)

FIG. 34

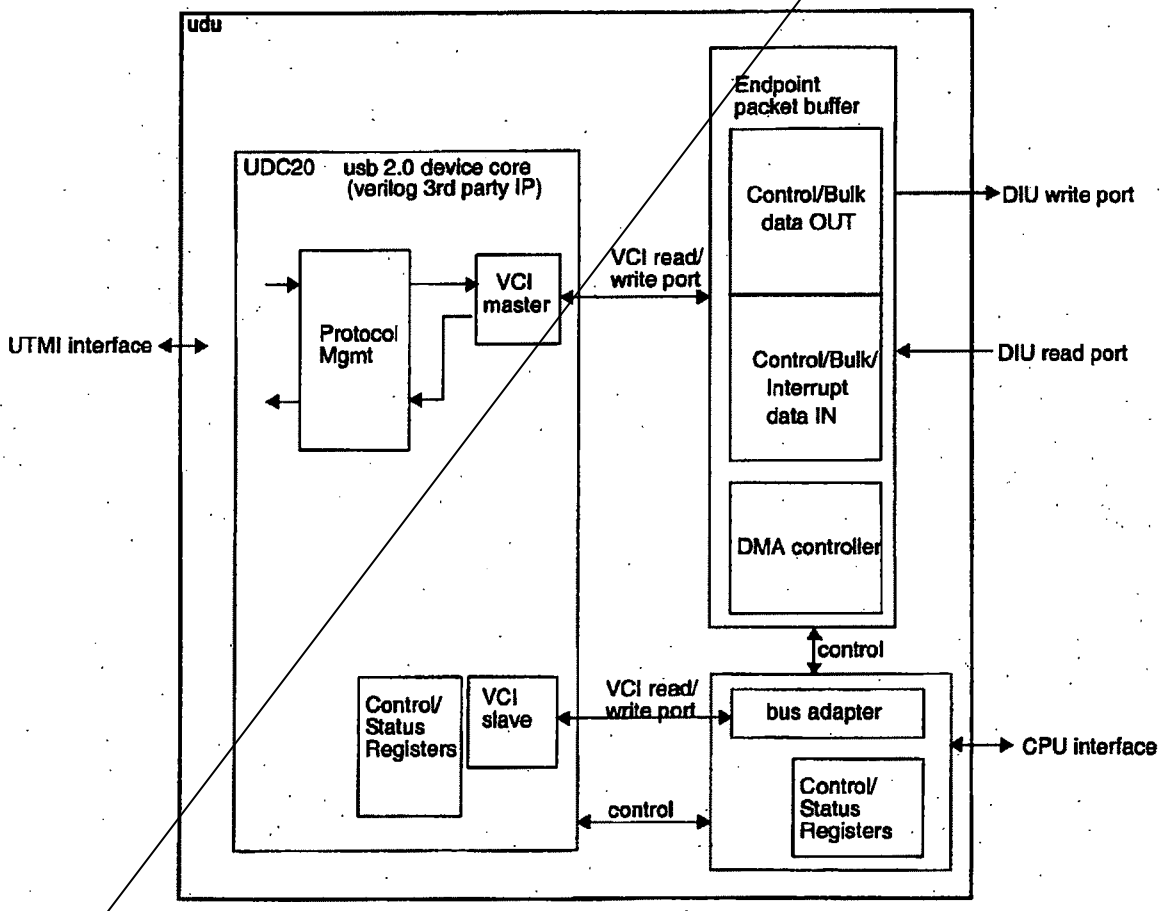

udu
UDC20 usb 2.0 device core (verilog 3rd party IP)
Protocol Mgmt
VCI master
UTMI interface
VCI read/ write port
Endpoint packet buffer
Control/Bulk data OUT
Control/Bulk/ Interrupt data IN
DMA controller
DIU write port
DIU read port
control
Control/ Status Registers
VCI slave
VCI read/ write port
bus adapter
CPU interface
control
Control/ Status Registers

FIG. 35

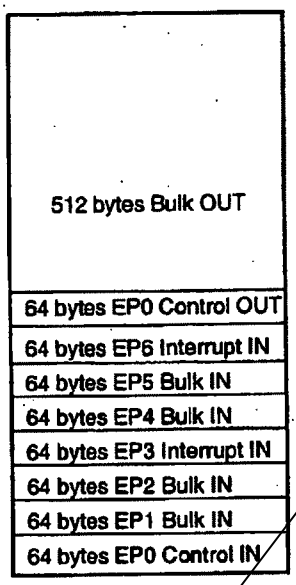

512 bytes Bulk OUT
64 bytes EP0 Control OUT
64 bytes EP6 Interrupt IN
64 bytes EP5 Bulk IN
64 bytes EP4 Bulk IN
64 bytes EP3 Interrupt IN
64 bytes EP2 Bulk IN
64 bytes EP1 Bulk IN
64 bytes EP0 Control IN

FIG. 36

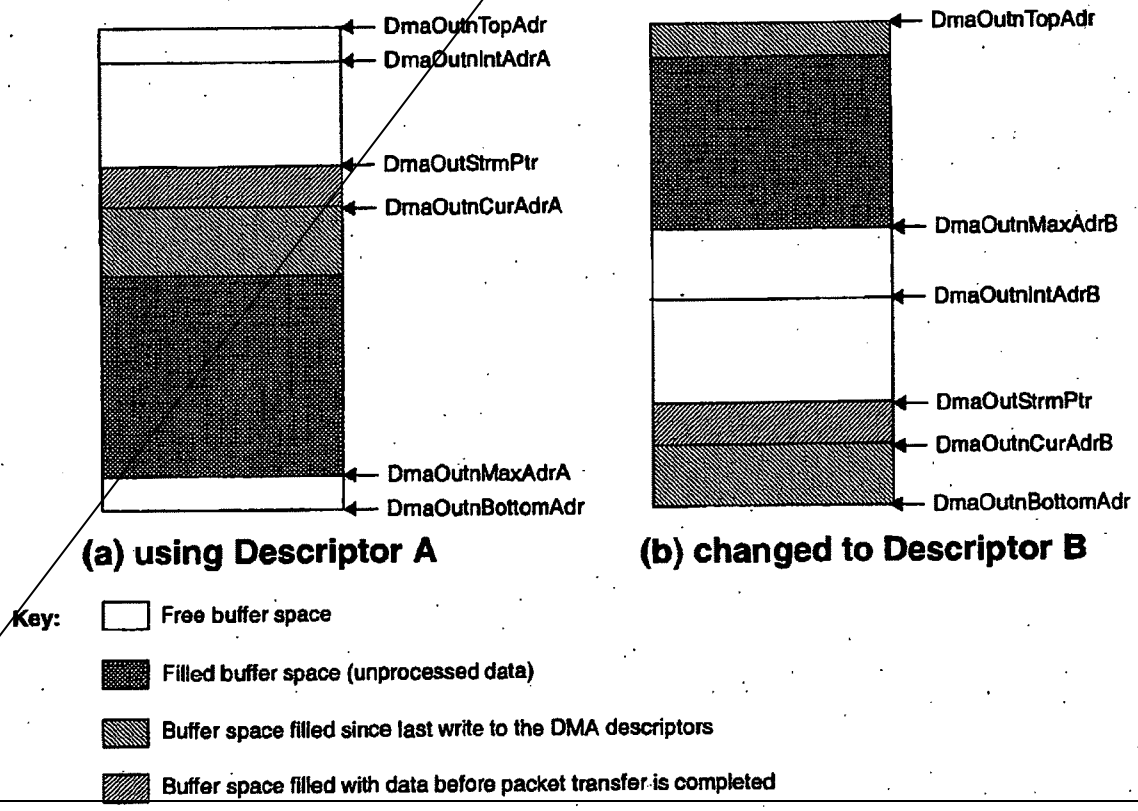

DmaOutnTopAdr
DmaOutnIntAdrA
DmaOutStrmPtr
DmaOutnCurAdrA
DmaOutnMaxAdrA
DmaOutnBottomAdr
(a) using Descriptor A
DmaOutnTopAdr
DmaOutnMaxAdrB
DmaOutnIntAdrB
DmaOutStrmPtr
DmaOutnCurAdrB
DmaOutnBottomAdr
(b) changed to Descriptor B
Key:
Free buffer space
Filled buffer space (unprocessed data)
Buffer space filled since last write to the DMA descriptors
Buffer space filled with data before packet transfer is completed

FIG. 37

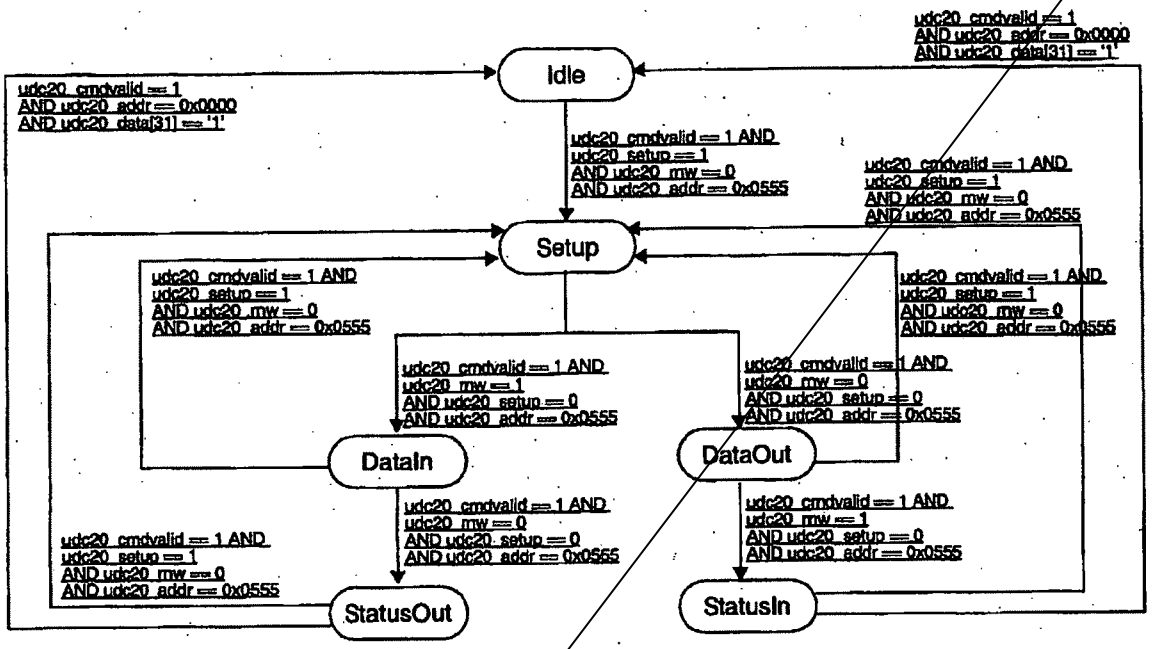

Idle
Setup
DataIn
DataOut
StatusOut
StatusIn
udc20_cmdvalid == 1
AND udc20_addr == 0x0000
AND udc20_data[31] == '1'
udc20_cmdvalid == 1 AND
udc20_setup == 1
AND udc20_rnw == 0
AND udc20_addr == 0x0555
udc20_cmdvalid == 1 AND
udc20_rnw == 1
AND udc20_setup == 0
AND udc20_addr == 0x0555
udc20_cmdvalid == 1 AND
udc20_rnw == 0
AND udc20_setup == 0
AND udc20_addr == 0x0555
udc20_cmdvalid == 1 AND
udc20_rnw == 0
AND udc20_setup == 0
AND udc20_addr == 0x0555
udc20_cmdvalid == 1 AND
udc20_rnw == 1
AND udc20_setup == 0
AND udc20_addr == 0x0555
udc20_cmdvalid == 1 AND
udc20_setup == 1
AND udc20_rnw == 0
AND udc20_addr == 0x0555

FIG. 38

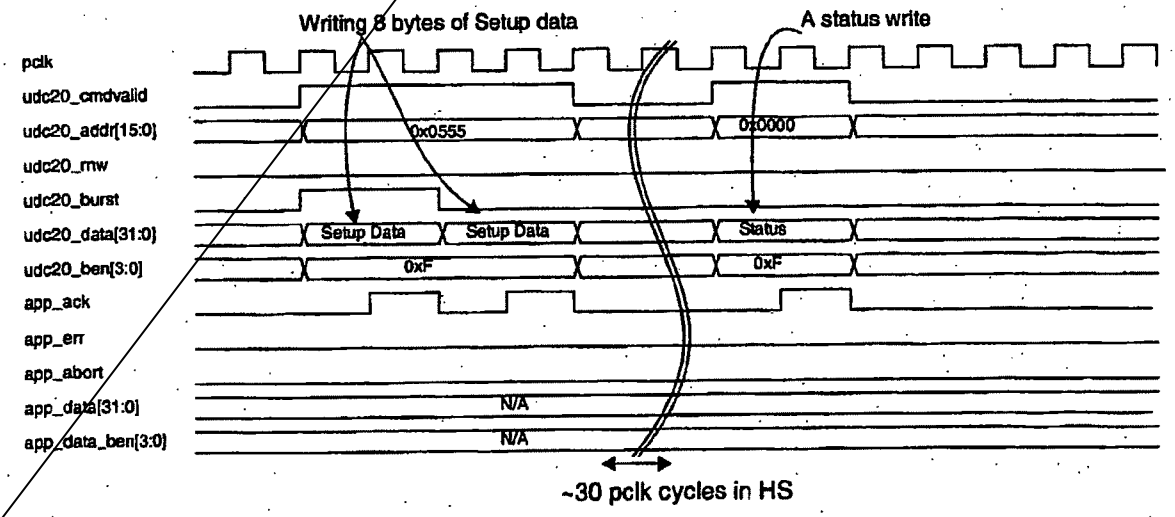

Writing 8 bytes of Setup data
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x0555
0x0000
Setup Data
Setup Data
Status
0xF
0xF
N/A
N/A
~30 pclk cycles in HS

FIG. 39

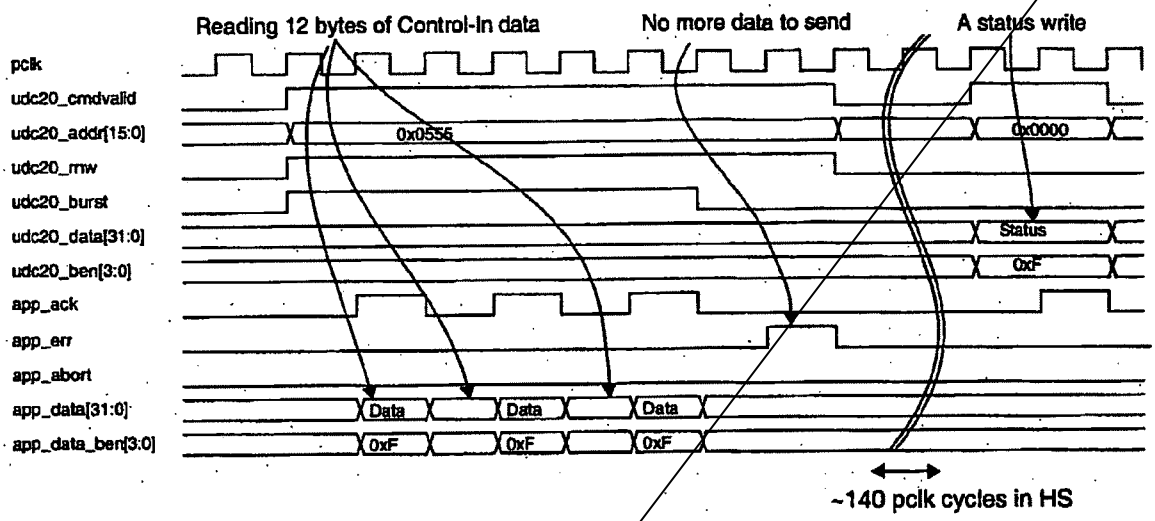

Reading 12 bytes of Control-In data
No more data to send
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x0556
0x0000
Status
0xF
Data
Data
Data
0xF
0xF
0xF
~140 pclk cycles in HS

FIG. 40

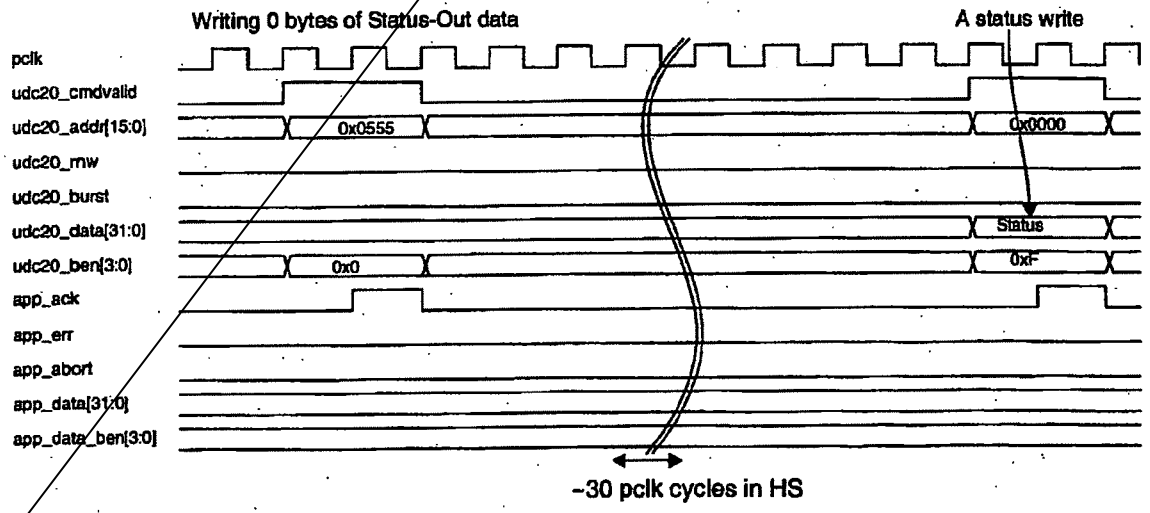

Writing 0 bytes of Status-Out data
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x0555
0x0000
Status
0xF
0x0
~30 pclk cycles in HS

FIG. 41

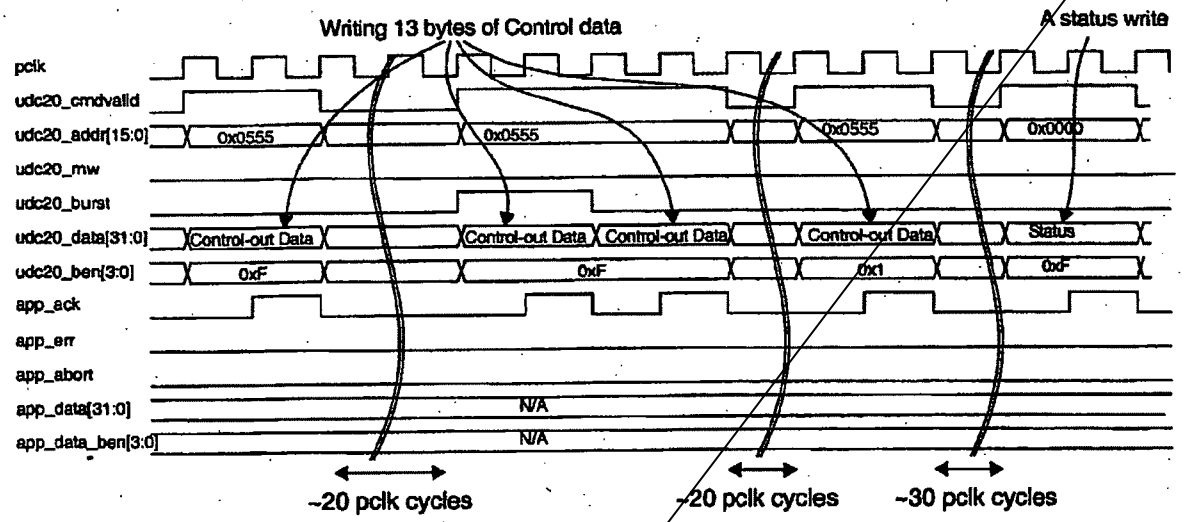

Writing 13 bytes of Control data
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x0555
0x0555
0x0555
0x0000
Control-out Data
Control-out Data
Control-out Data
Control-out Data
Status
0xF
0xF
0x1
0xF
N/A
N/A
~20 pclk cycles
~20 pclk cycles
~30 pclk cycles

FIG. 42

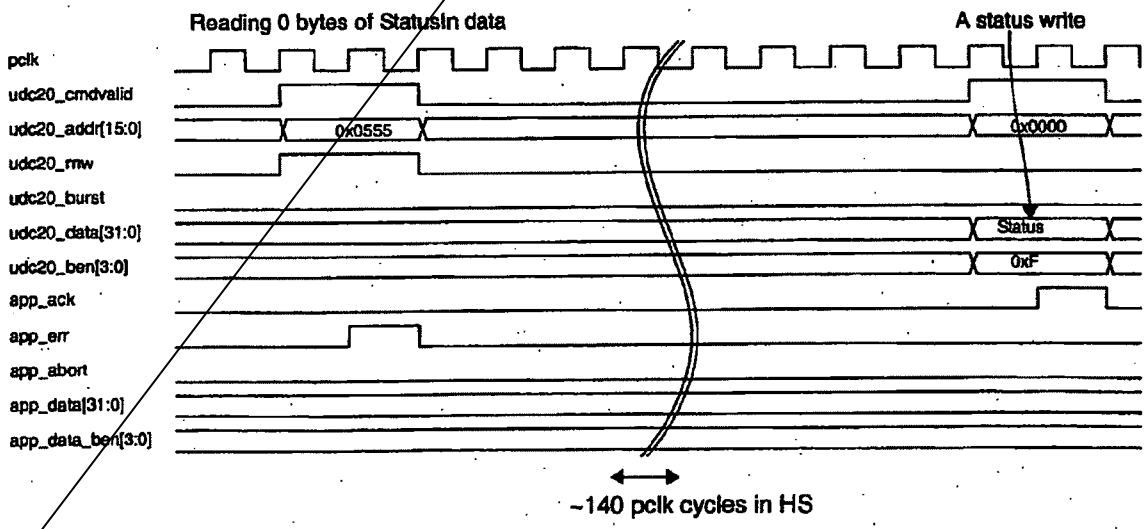

Reading 0 bytes of StatusIn data
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x0555
0x0000
Status
0xF
~140 pclk cycles in HS

FIG. 43

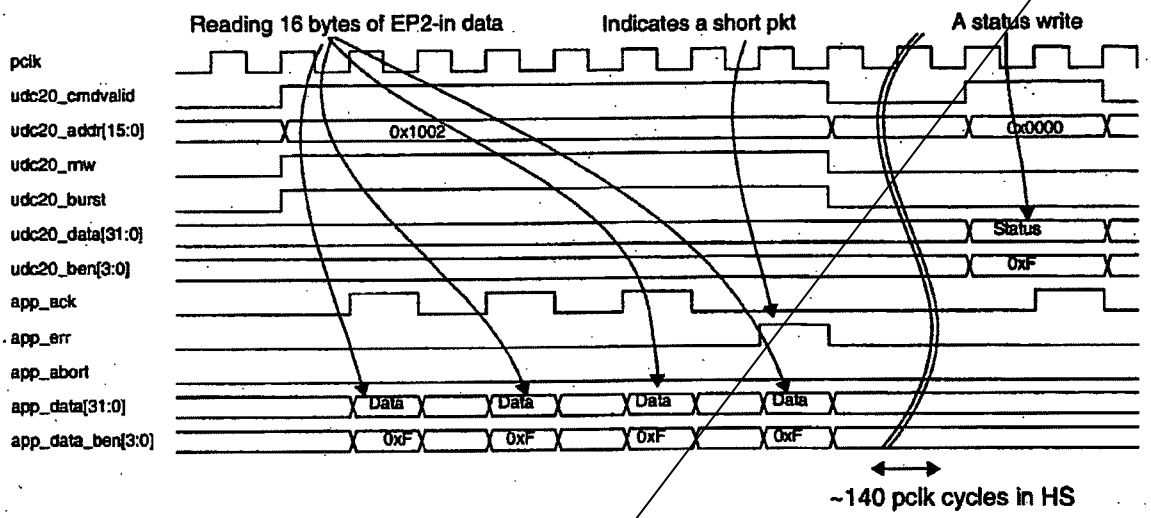

Reading 16 bytes of EP2-in data
Indicates a short pkt
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x1002
0x0000
Status
0xF
Data
0xF
~140 pclk cycles in HS

FIG. 44

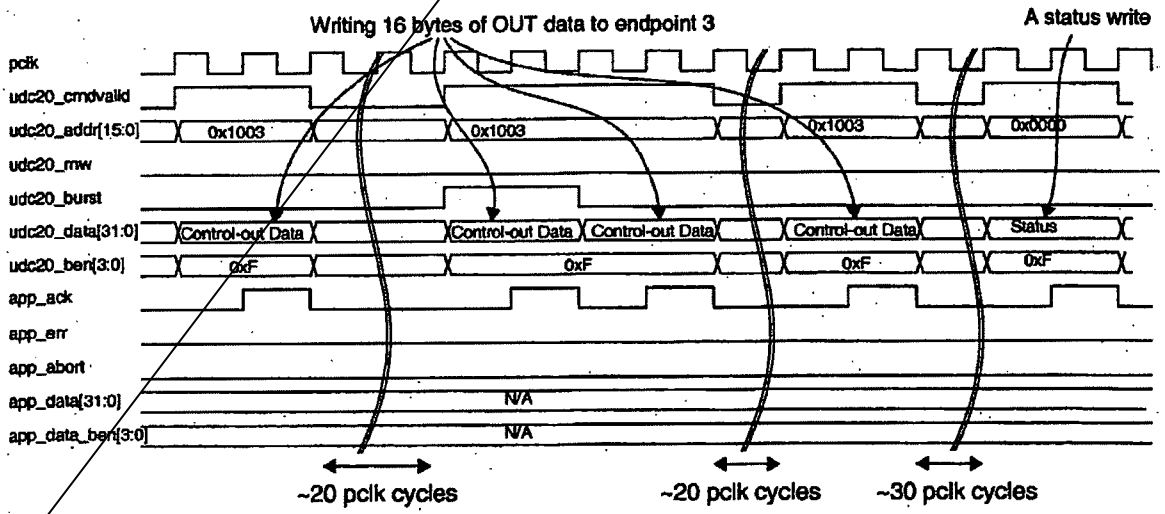

Writing 16 bytes of OUT data to endpoint 3
A status write
pclk
udc20_cmdvalid
udc20_addr[15:0]
udc20_rnw
udc20_burst
udc20_data[31:0]
udc20_ben[3:0]
app_ack
app_err
app_abort
app_data[31:0]
app_data_ben[3:0]
0x1003
0x0000
Control-out Data
Status
0xF
N/A
~20 pclk cycles
~20 pclk cycles
~30 pclk cycles

FIG. 45

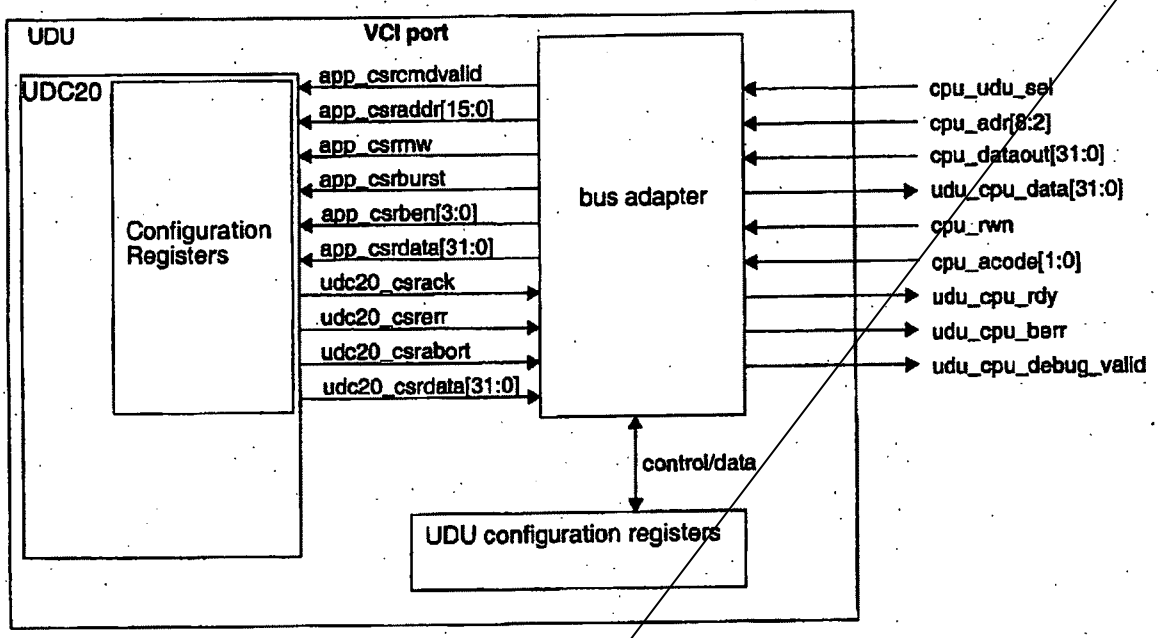


FIG. 46

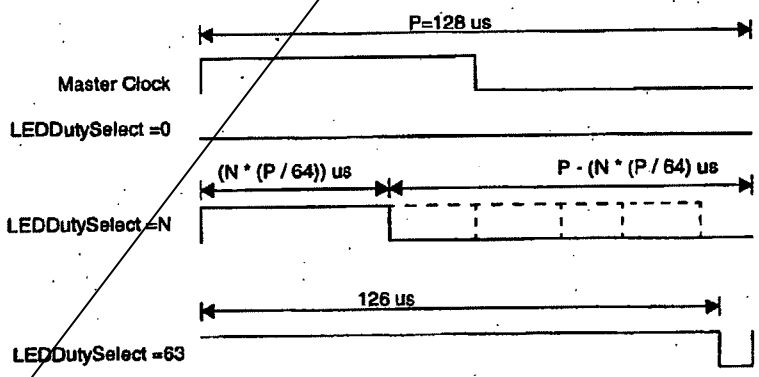


FIG. 47

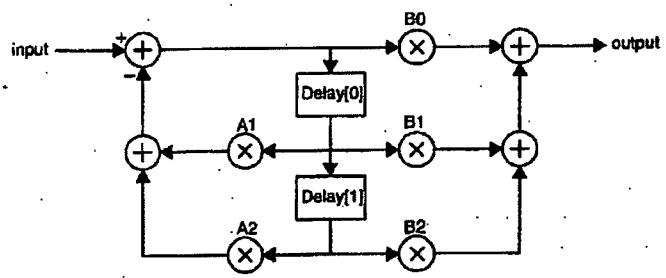


FIG. 48

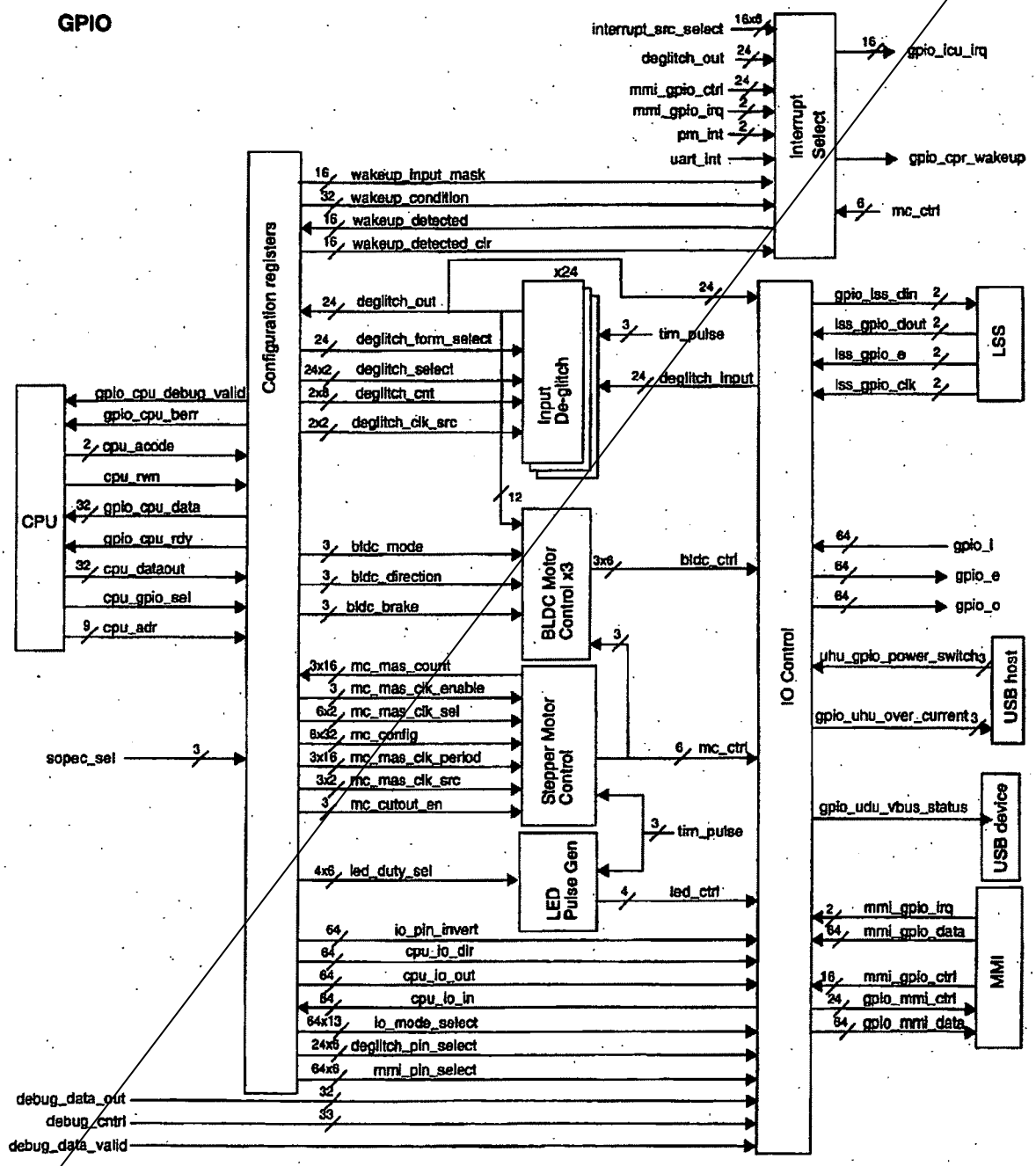

GPIO
interrupt_src_select
deglitch_out
mmi_gpio_ctrl
mmi_gpio_irq
pm_int
uart_int
Interrupt Select
gpio_icu_irq
gpio_cpr_wakeup
mc_ctrl
wakeup_input_mask
wakeup_condition
wakeup_detected
wakeup_detected_clr
Configuration registers
deglitch_out
deglitch_form_select
deglitch_select
deglitch_cnt
deglitch_clk_src
Input De-glitch
tim_pulse
deglitch_input
gpio_lss_din
lss_gpio_dout
lss_gpio_e
lss_gpio_clk
LSS
CPU
gpio_cpu_debug_valid
gpio_cpu_berr
cpu_acode
cpu_rwn
gpio_cpu_data
gpio_cpu_rdy
cpu_dataout
cpu_gpio_sel
cpu_adr
BLDC Motor Control x3
bldc_mode
bldc_direction
bldc_brake
bldc_ctrl
gpio_i
gpio_e
gpio_o
IO Control
uhu_gpio_power_switch3
gpio_uhu_over_current3
USB host
sopec_sel
Stepper Motor Control
mc_mas_count
mc_mas_clk_enable
mc_mas_clk_sel
mc_config
mc_mas_clk_period
mc_mas_clk_src
mc_cutout_en
mc_ctrl
gpio_udu_vbus_status
USB device
LED Pulse Gen
led_duty_sel
led_ctrl
io_pin_invert
cpu_io_dir
cpu_io_out
cpu_io_in
io_mode_select
deglitch_pin_select
mmi_pin_select
mmi_gpio_irq
mmi_gpio_data
mmi_gpio_ctrl
gpio_mmi_ctrl
gpio_mmi_data
MMI
debug_data_out
debug_cntrl
debug_data_valid

FIG. 49

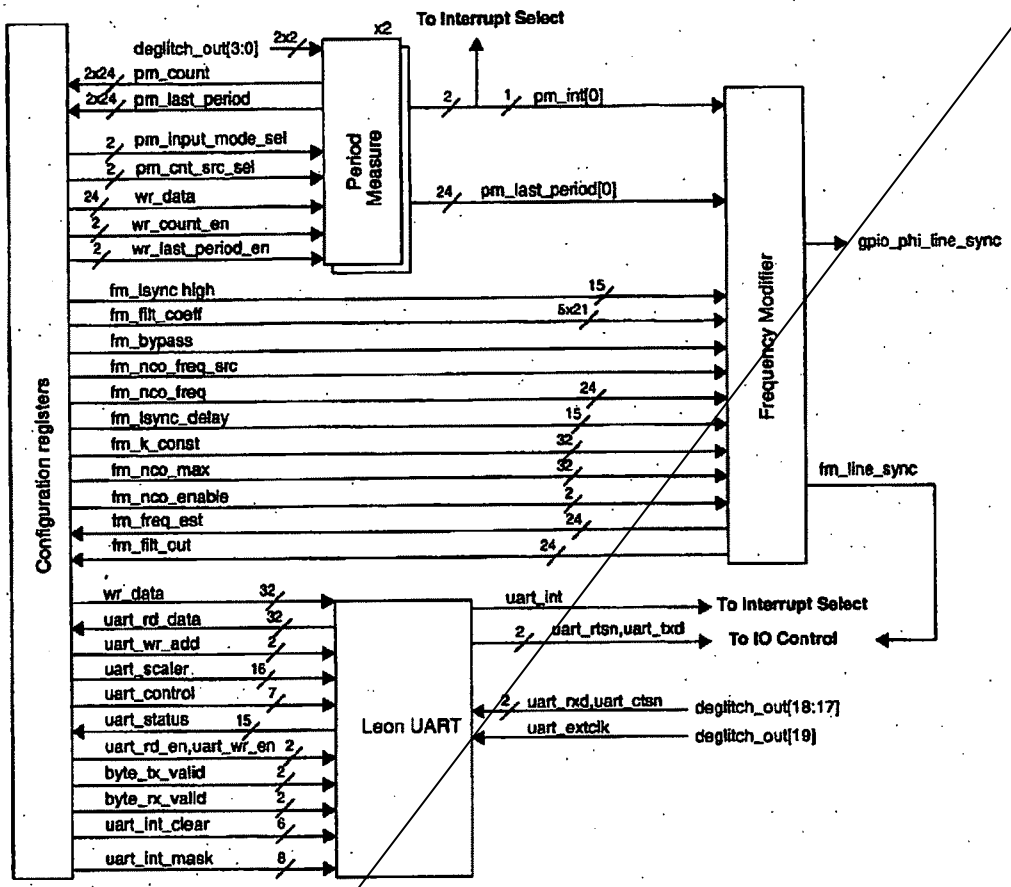

Configuration registers
deglitch_out[3:0]
2x2
x2
To Interrupt Select
2x24 pm_count
2x24 pm_last_period
2 pm_input_mode_sel
2 pm_cnt_src_sel
24 wr_data
2 wr_count_en
2 wr_last_period_en
Period Measure
2
1 pm_int[0]
24 pm_last_period[0]
Frequency Modifier
gpio_phi_line_sync
fm_lsync high
15
fm_filt_coeff
5x21
fm_bypass
fm_nco_freq_src
fm_nco_freq
24
fm_lsync_delay
15
fm_k_const
32
fm_nco_max
32
fm_nco_enable
2
fm_freq_est
24
fm_filt_out
24
fm_line_sync
wr_data
32
uart_rd_data
32
uart_wr_add
2
uart_scaler
16
uart_control
7
uart_status
15
uart_rd_en,uart_wr_en
2
byte_tx_valid
2
byte_rx_valid
2
uart_int_clear
8
uart_int_mask
8
Leon UART
uart_int
To Interrupt Select
2 uart_rtsn,uart_txd
To IO Control
2 uart_rxd,uart_ctsn
deglitch_out[18:17]
uart_extclk
deglitch_out[19]

FIG. 50

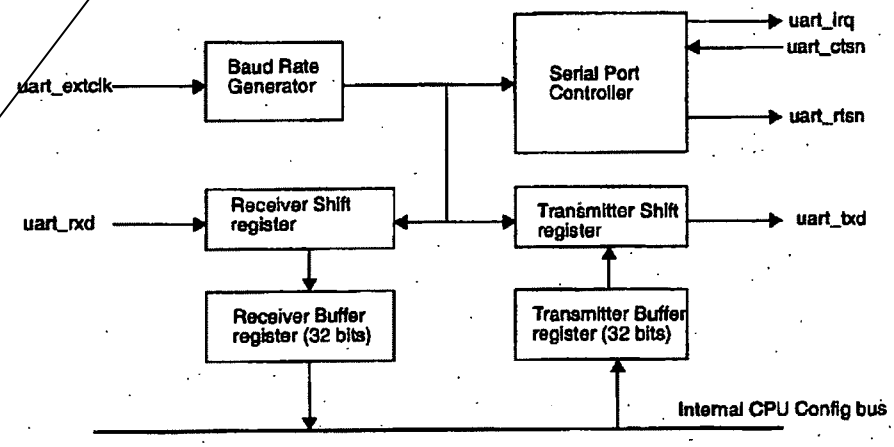

uart_extclk
Baud Rate Generator
Serial Port Controller
uart_irq
uart_ctsn
uart_rtsn
uart_rxd
Receiver Shift register
Transmitter Shift register
uart_txd
Receiver Buffer register (32 bits)
Transmitter Buffer register (32 bits)
Internal CPU Config bus

FIG. 51

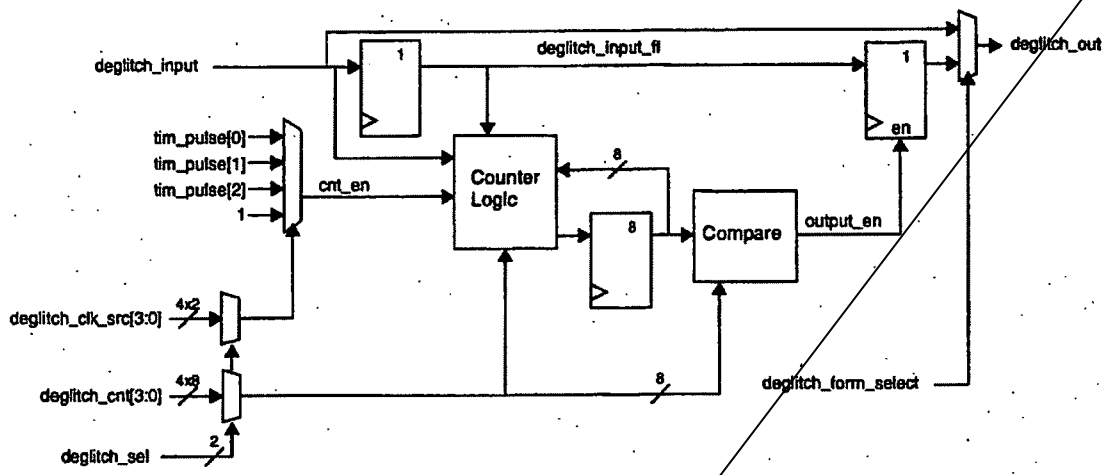


FIG. 52

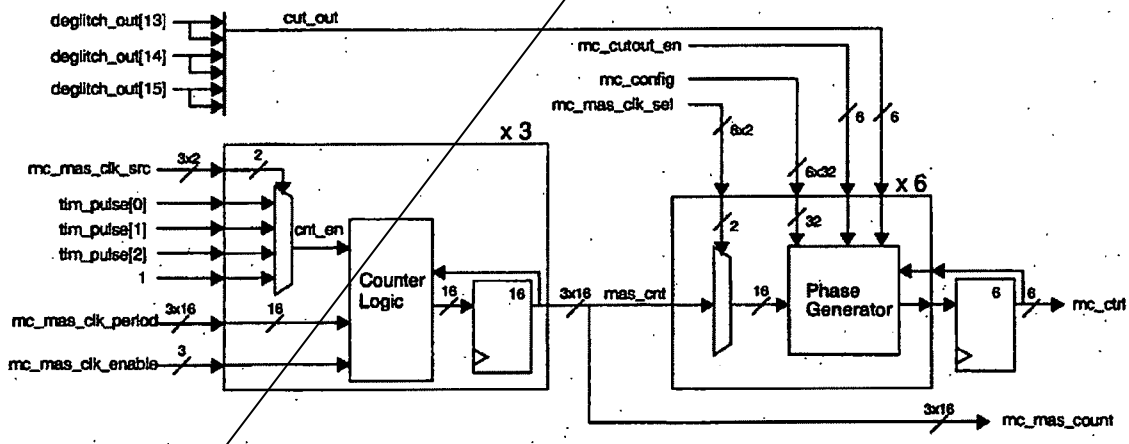


FIG. 53

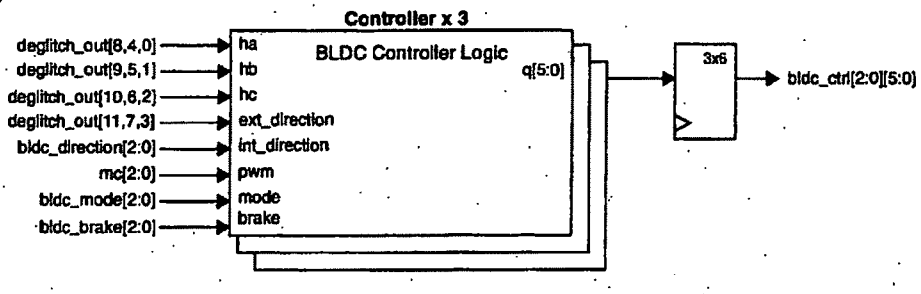


FIG. 54

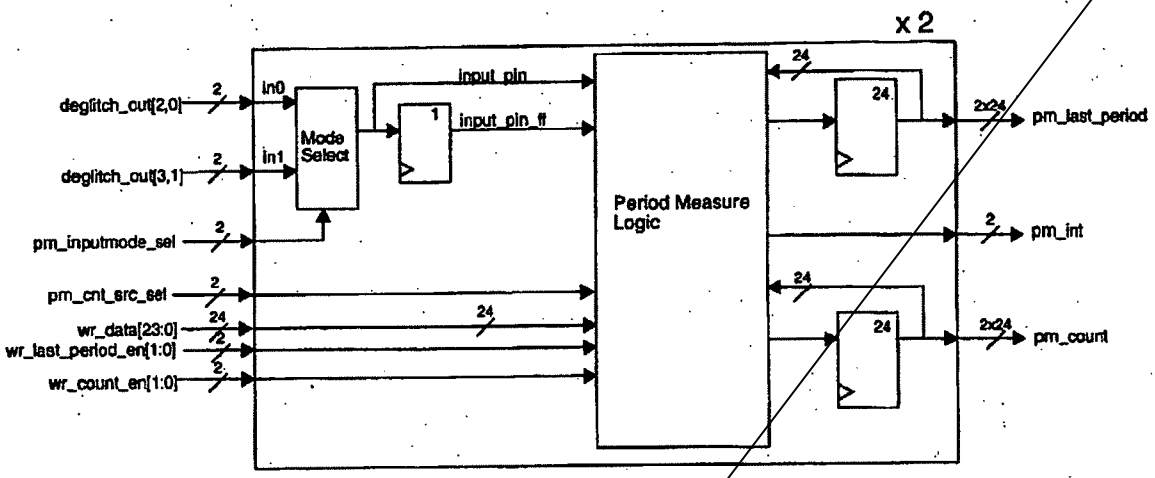


FIG. 55

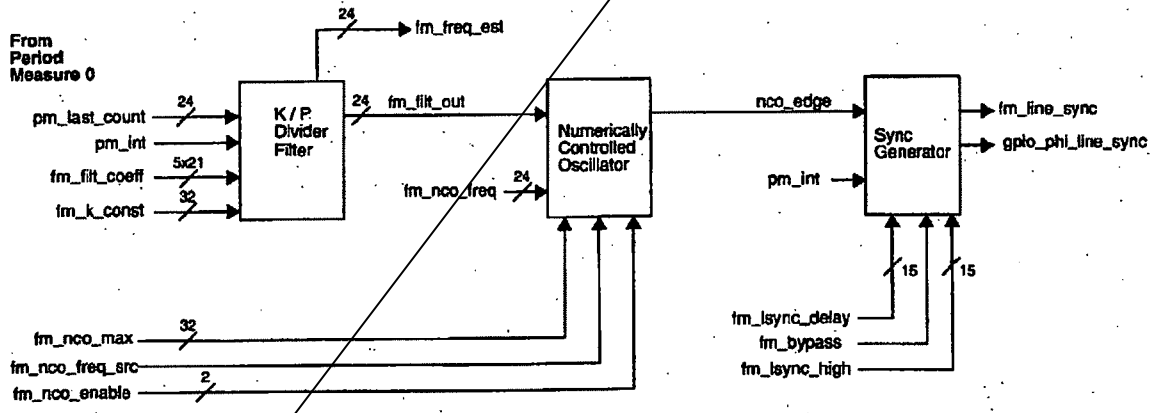


FIG. 56

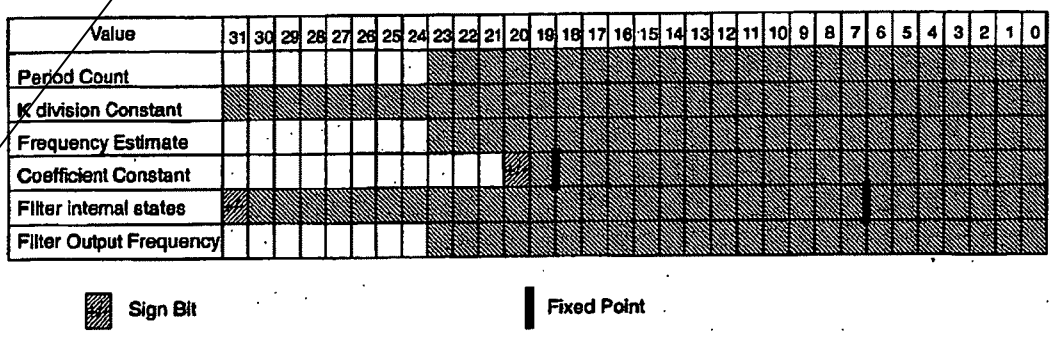


FIG. 57

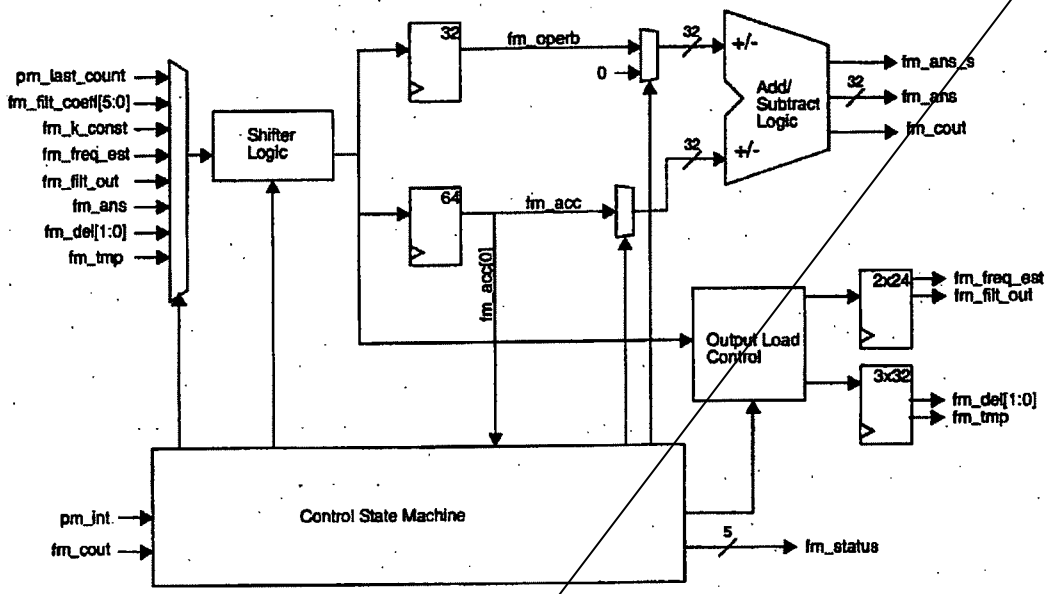

pm_last_count
fm_filt_coeff[5:0]
fm_k_const
fm_freq_est
fm_filt_out
fm_ans
fm_del[1:0]
fm_tmp
Shifter Logic
32
fm_operb
0
64
fm_acc
fm_acc[0]
+/-
Add/ Subtract Logic
fm_ans_s
fm_ans
fm_cout
Output Load Control
2x24
fm_freq_est
fm_filt_out
3x32
fm_del[1:0]
fm_tmp
pm_int
fm_cout
Control State Machine
5
fm_status

FIG. 58

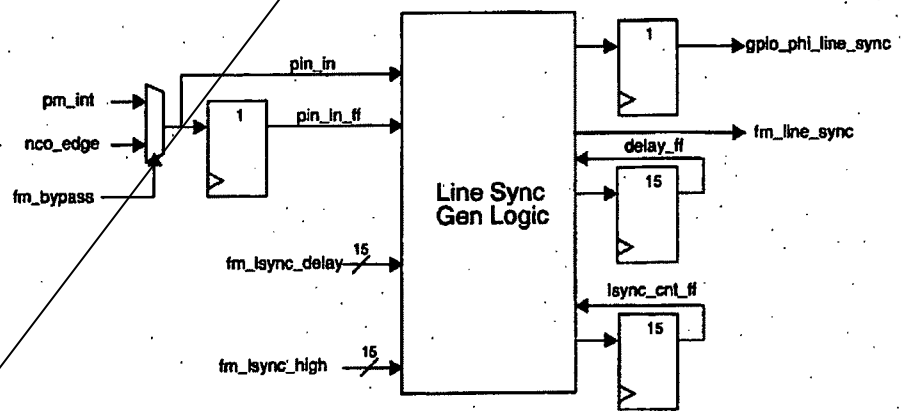

pm_int
nco_edge
fm_bypass
pin_in
pin_in_ff
1
fm_lsync_delay
15
fm_lsync_high
15
Line Sync Gen Logic
1
gpio_phi_line_sync
fm_line_sync
delay_ff
15
lsync_cnt_ff
15

FIG. 59

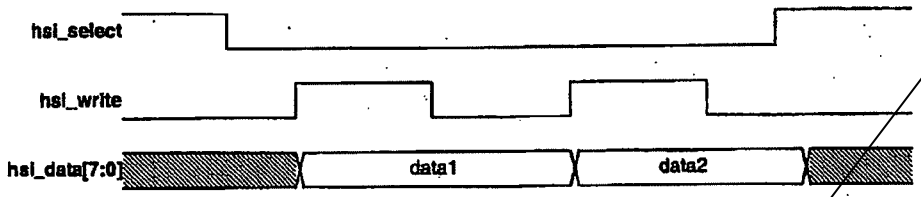


FIG. 60

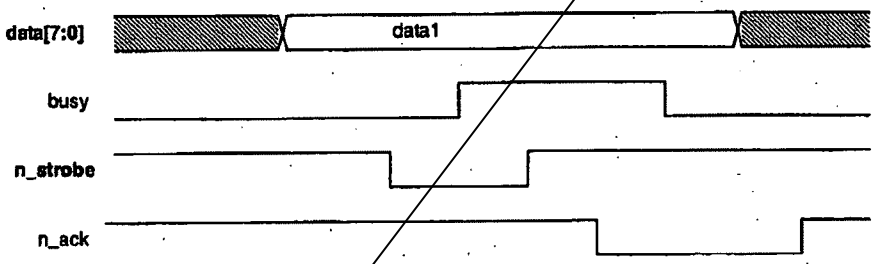


FIG. 61

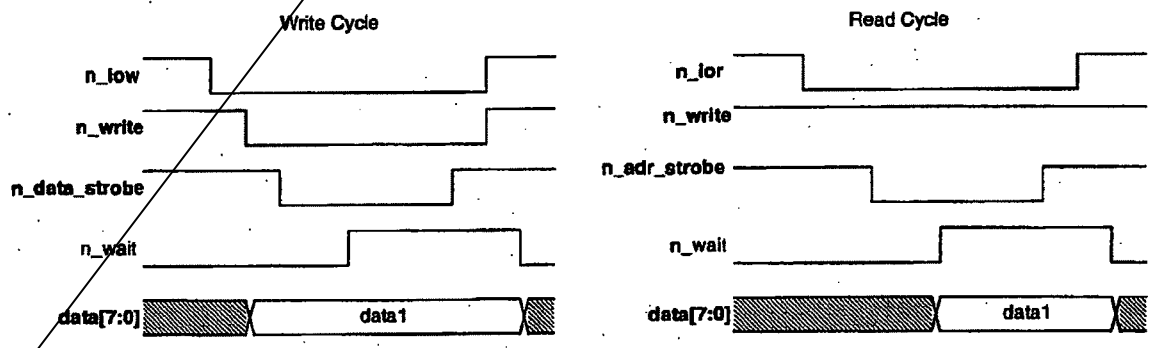


FIG. 62

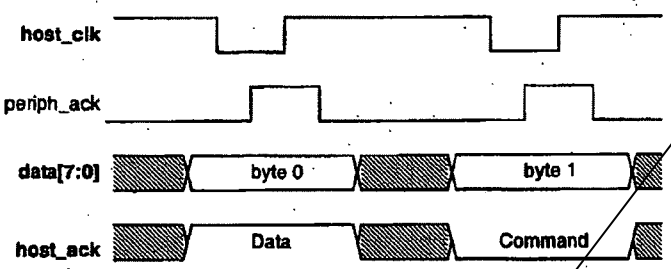

host_clk
periph_ack
data[7:0]
byte 0
byte 1
host_ack
Data
Command

FIG. 63

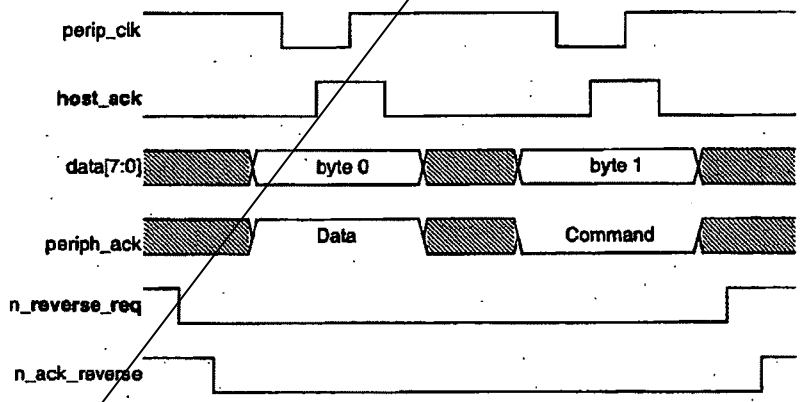

perip_clk
host_ack
data[7:0]
byte 0
byte 1
periph_ack
Data
Command
n_reverse_req
n_ack_reverse

FIG. 64

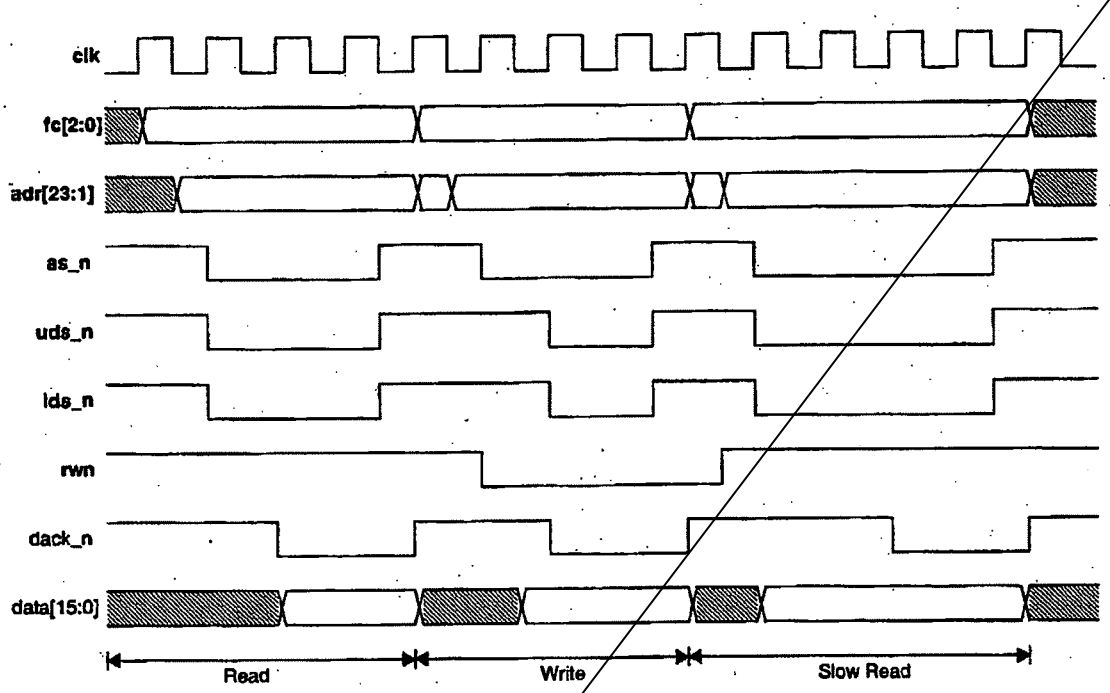

clk
fc[2:0]
adr[23:1]
as_n
uds_n
lds_n
rwn
dack_n
data[15:0]
Read
Write
Slow Read

FIG. 65

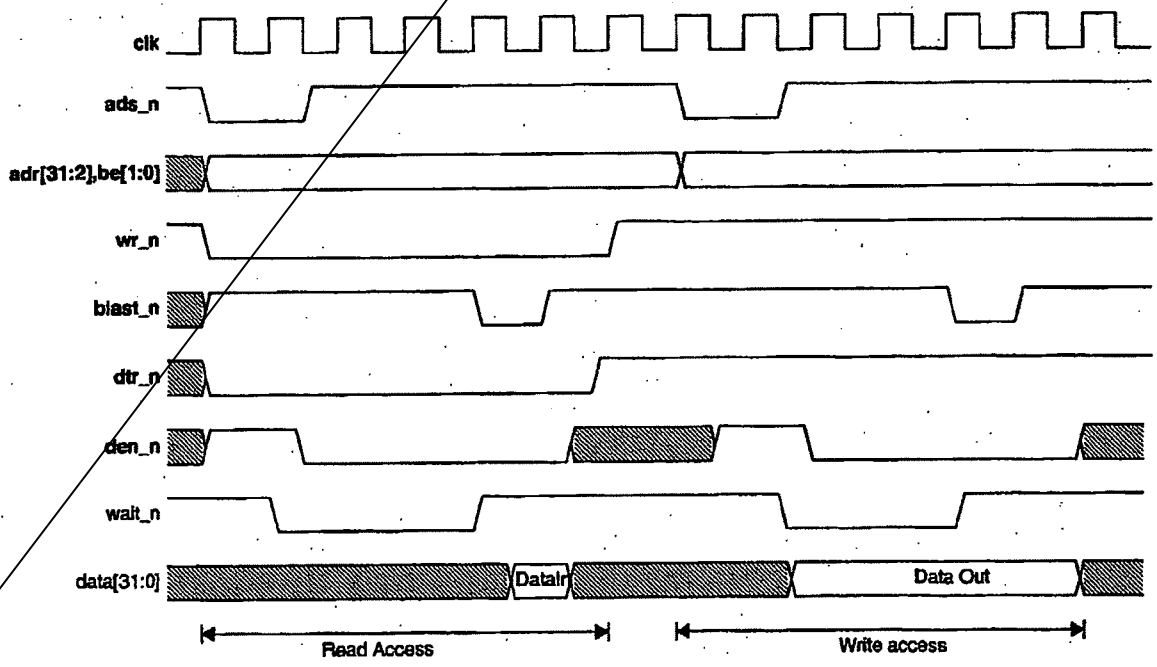

clk
ads_n
adr[31:2],be[1:0]
wr_n
blast_n
dtr_n
den_n
wait_n
data[31:0]
DataIn
Data Out
Read Access
Write access

FIG. 66

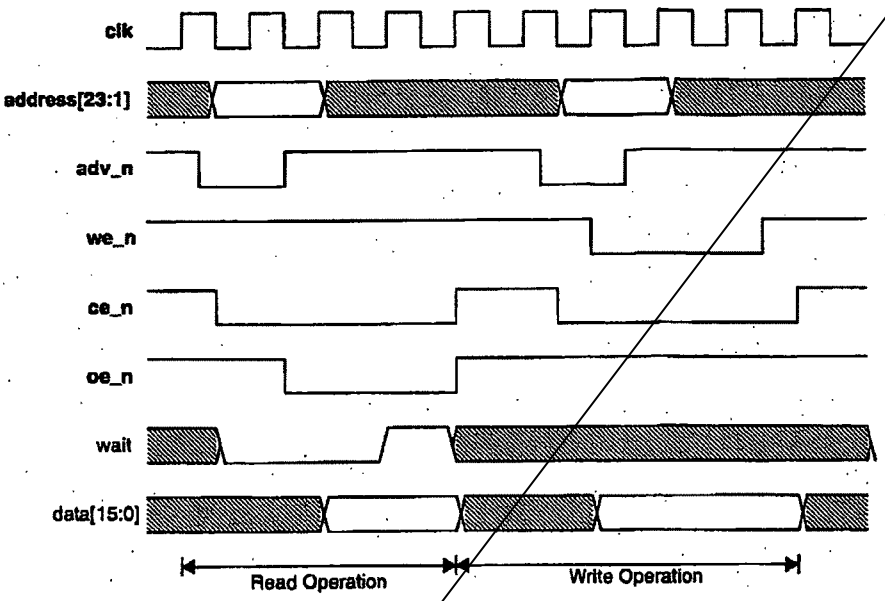

clk
address[23:1]
adv_n
we_n
ce_n
oe_n
wait
data[15:0]
Read Operation
Write Operation

FIG. 67

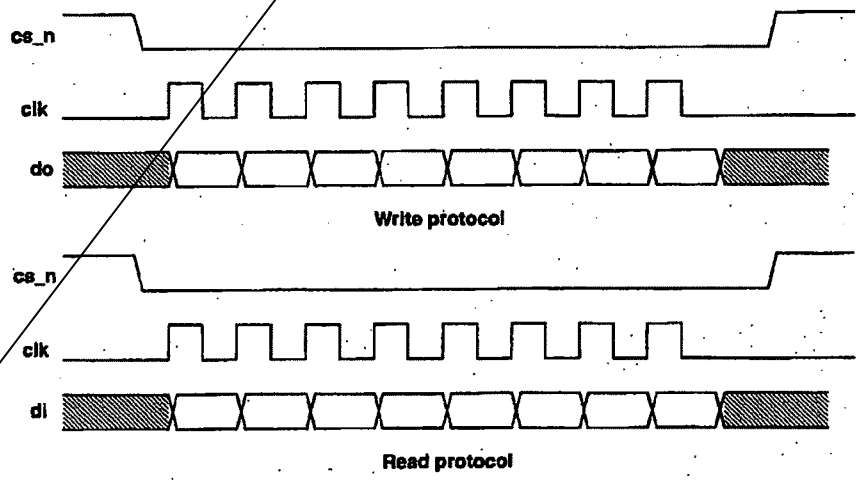

cs_n
clk
do
Write protocol
cs_n
clk
di
Read protocol

FIG. 68

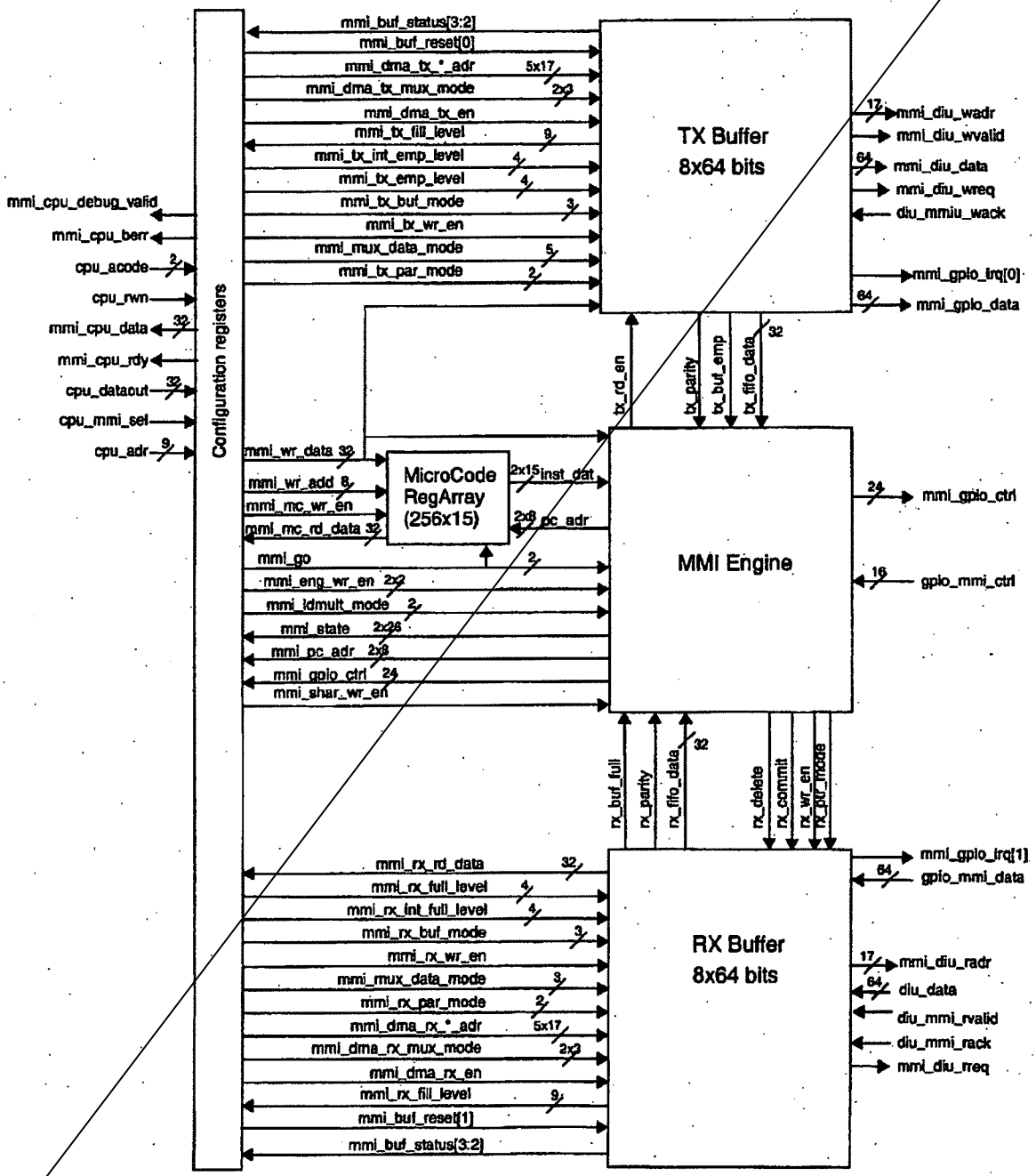

Configuration registers
mmi_cpu_debug_valid
mmi_cpu_berr
cpu_acode
cpu_rwn
mmi_cpu_data
mmi_cpu_rdy
cpu_dataout
cpu_mmi_sel
cpu_adr
mmi_buf_status[3:2]
mmi_buf_reset[0]
mmi_dma_tx_*_adr
mmi_dma_tx_mux_mode
mmi_dma_tx_en
mmi_tx_fill_level
mmi_tx_int_emp_level
mmi_tx_emp_level
mmi_tx_buf_mode
mmi_tx_wr_en
mmi_mux_data_mode
mmi_tx_par_mode
TX Buffer
8x64 bits
mmi_diu_wadr
mmi_diu_wvalid
mmi_diu_data
mmi_diu_wreq
diu_mmiu_wack
mmi_gpio_irq[0]
mmi_gpio_data
tx_rd_en
tx_parity
tx_buf_emp
tx_fifo_data
mmi_wr_data
mmi_wr_add
mmi_mc_wr_en
mmi_mc_rd_data
MicroCode
RegArray
(256x15)
inst_dat
pc_adr
mmi_go
mmi_eng_wr_en
mmi_ldmult_mode
mmi_state
mmi_pc_adr
mmi_gpio_ctrl
mmi_shar_wr_en
MMI Engine
mmi_gpio_ctrl
gpio_mmi_ctrl
rx_buf_full
rx_parity
rx_fifo_data
rx_delete
rx_commit
rx_wr_en
rx_put_mode
mmi_rx_rd_data
mmi_rx_full_level
mmi_rx_int_full_level
mmi_rx_buf_mode
mmi_rx_wr_en
mmi_mux_data_mode
mmi_rx_par_mode
mmi_dma_rx_*_adr
mmi_dma_rx_mux_mode
mmi_dma_rx_en
mmi_rx_fill_level
mmi_buf_reset[1]
mmi_buf_status[3:2]
RX Buffer
8x64 bits
mmi_gpio_irq[1]
gpio_mmi_data
mmi_diu_radr
diu_data
diu_mmi_rvalid
diu_mmi_rack
mmi_diu_rreq

FIG. 69

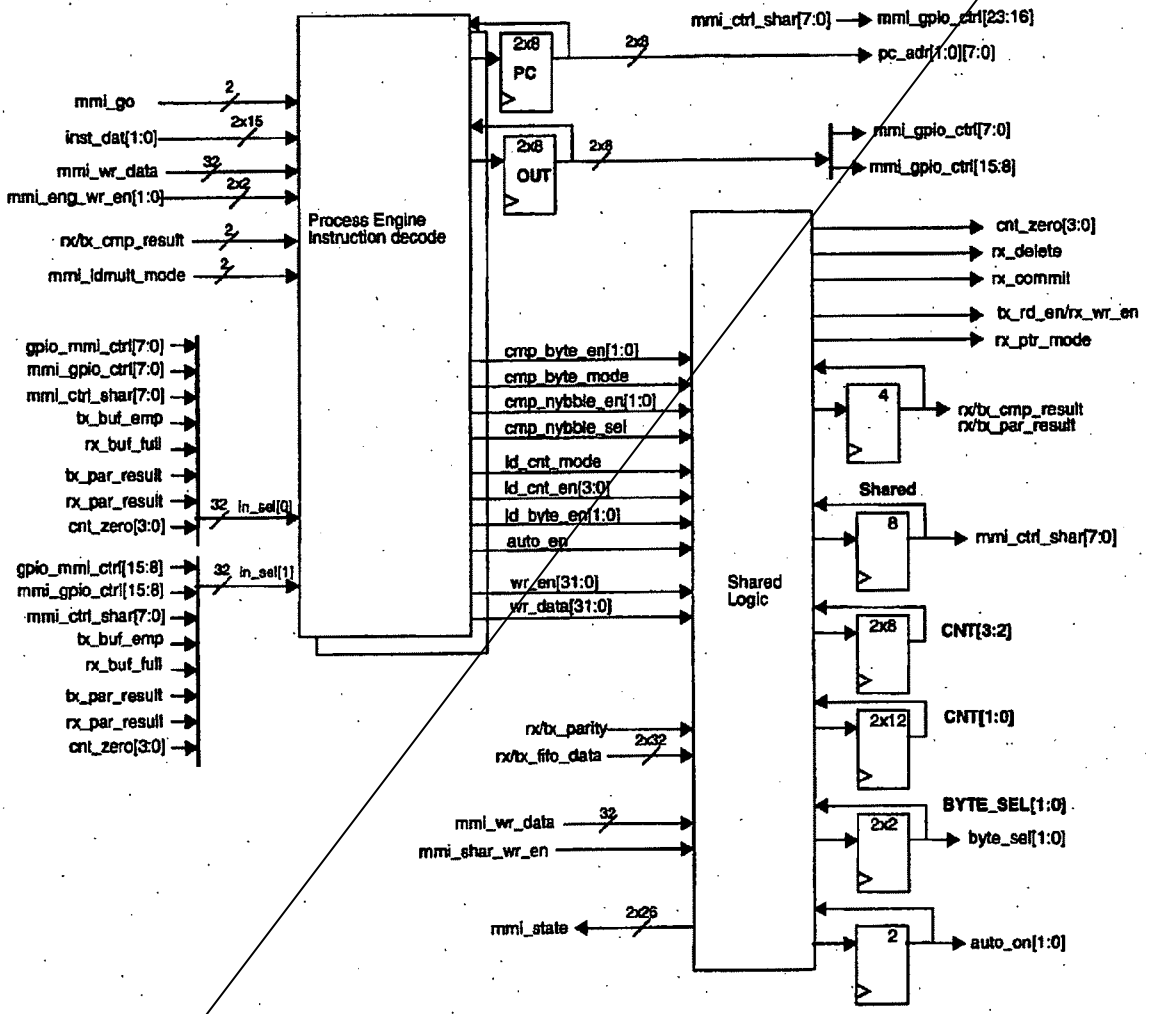

mmi_ctrl_shar[7:0]
mmi_gpio_ctrl[23:16]
2x8
PC
2x8
pc_adr[1:0][7:0]
mmi_go
2
inst_dat[1:0]
2x15
mmi_wr_data
32
mmi_eng_wr_en[1:0]
2x2
rx/tx_cmp_result
2
mmi_ldmult_mode
2
Process Engine
Instruction decode
2x8
OUT
2x8
mmi_gpio_ctrl[7:0]
mmi_gpio_ctrl[15:8]
cnt_zero[3:0]
rx_delete
rx_commit
tx_rd_en/rx_wr_en
rx_ptr_mode
gpio_mmi_ctrl[7:0]
mmi_gpio_ctrl[7:0]
mmi_ctrl_shar[7:0]
tx_buf_emp
rx_buf_full
tx_par_result
rx_par_result
cnt_zero[3:0]
32
in_sel[0]
cmp_byte_en[1:0]
cmp_byte_mode
cmp_nybble_en[1:0]
cmp_nybble_sel
ld_cnt_mode
ld_cnt_en[3:0]
ld_byte_en[1:0]
auto_en
4
rx/tx_cmp_result
rx/tx_par_result
Shared
8
mmi_ctrl_shar[7:0]
gpio_mmi_ctrl[15:8]
mmi_gpio_ctrl[15:8]
mmi_ctrl_shar[7:0]
tx_buf_emp
rx_buf_full
tx_par_result
rx_par_result
cnt_zero[3:0]
32
in_sel[1]
wr_en[31:0]
wr_data[31:0]
Shared
Logic
2x8
CNT[3:2]
2x12
CNT[1:0]
rx/tx_parity
rx/tx_fifo_data
2x32
BYTE_SEL[1:0]
2x2
byte_sel[1:0]
mmi_wr_data
32
mmi_shar_wr_en
mmi_state
2x26
2
auto_on[1:0]

FIG. 70

Inst	Values
BC	1
ACNT	01
LDMULT	001
CMPNYBBLE	00011
CMYBYTE	00010
LDCNT	00001
BCCMP1	000001
LDBIT	000000001
LDCNT_FIFO	0000000001
LDBSEL	000000000001
RX_COMMIT	00000000000001
RX_DELETE	00000000000000

14 12 11 8 7 4 3 0	Instructions			
BC	B	IN_SEL	PC_OFFSET	
ACNT	B	NUM_CYCLES		
LDMULT	MASK	OUT_VALUE		
CMPNYBBLE	B2	B1	MASK	VALUE
CMPBYTE	B1	ME	MASK/DATA VALUE	
LDCNT	SEL	NUM_COUNT		
BCCMP1	B	PC_OFFSET		
LDBIT	B	OUT_SEL		
LDCNT_FIFO	B	SEL	NYB	
LDBSEL	B	SEL		
RX_COMMIT				
RX_DELETE				

FIG. 71

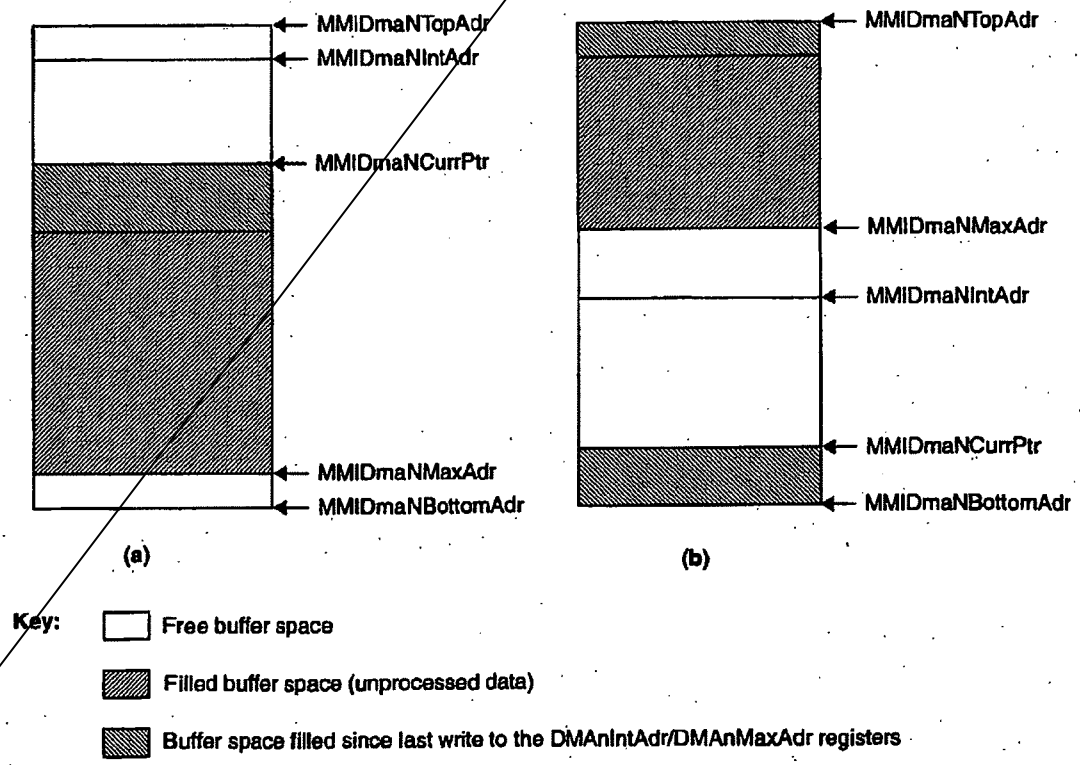


FIG. 72

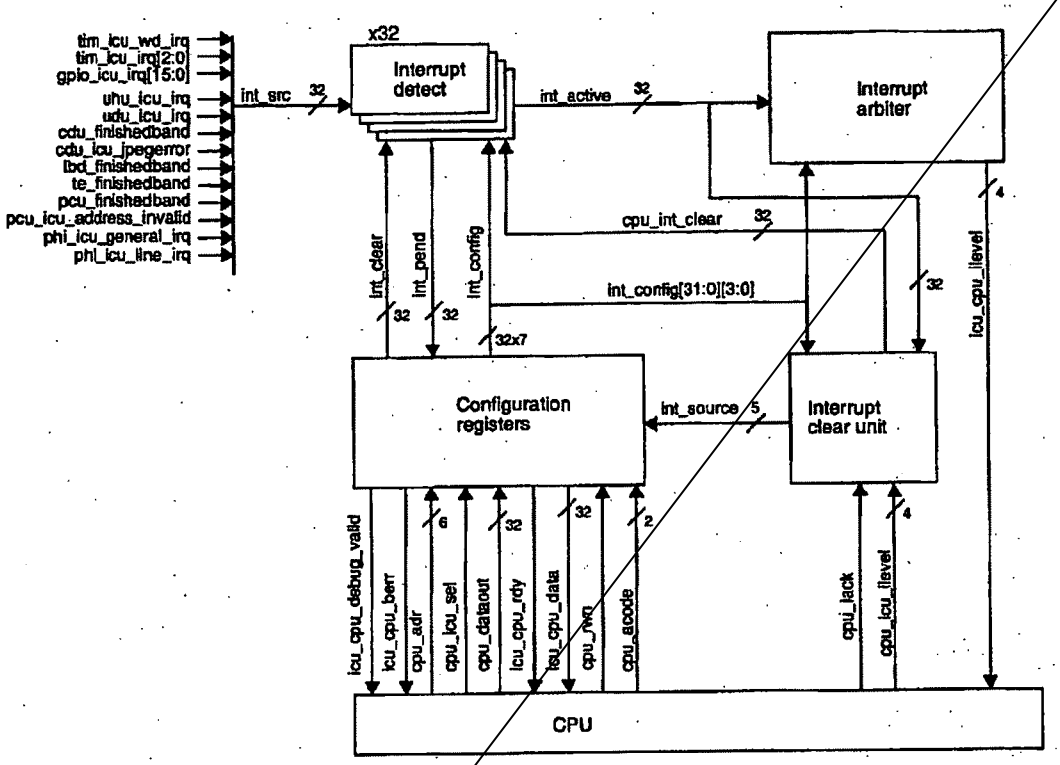

tim_icu_wd_irq
tim_icu_irq[2:0]
gpio_icu_irq[15:0]
uhu_icu_irq
udu_icu_irq
cdu_finishedband
cdu_icu_jpegerror
tbd_finishedband
te_finishedband
pcu_finishedband
pcu_icu_address_invalid
phi_icu_general_irq
phi_icu_line_irq
int_src
32
x32
Interrupt detect
int_active
32
Interrupt arbiter
cpu_int_clear
32
int_clear
32
int_pend
32
int_config
32x7
int_config[31:0][3:0]
32
icu_cpu_ilevel
4
Configuration registers
int_source
5
Interrupt clear unit
icu_cpu_debug_valid
icu_cpu_berr
cpu_adr
6
cpu_icu_sel
cpu_dataout
32
icu_cpu_rdy
icu_cpu_data
32
cpu_rwn
cpu_acode
2
cpu_iack
cpu_icu_ilevel
4
CPU

FIG. 73

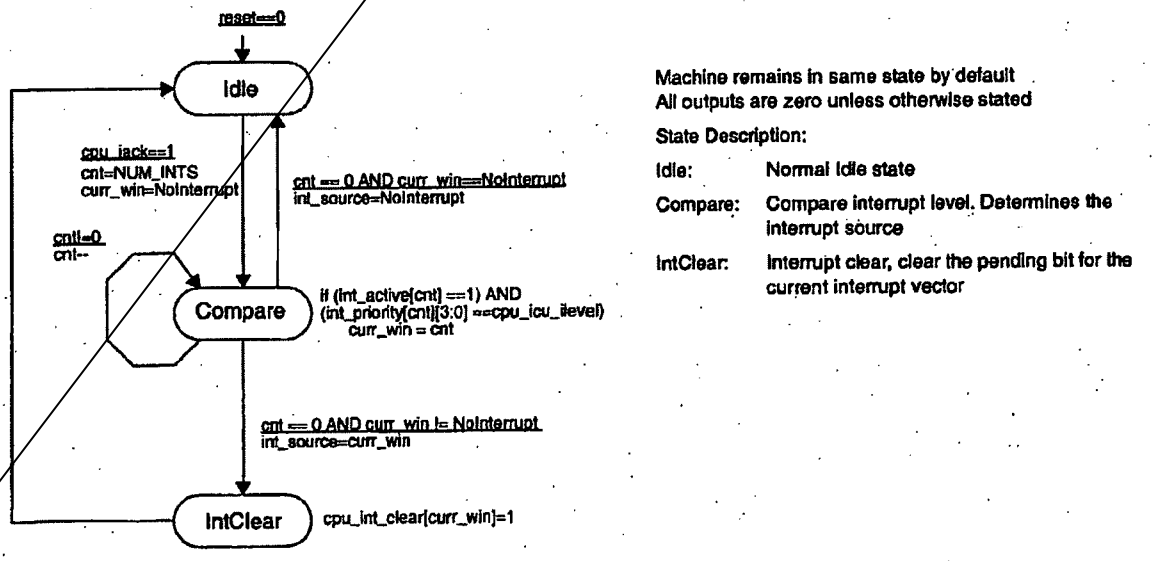

reset==0
Idle
cpu_iack==1
cnt=NUM_INTS
curr_win=NoInterrupt
cnt == 0 AND curr_win==NoInterrupt
int_source=NoInterrupt
cnt!=0
cnt--
Compare
if (int_active[cnt] ==1) AND
(int_priority[cnt][3:0] ==cpu_icu_ilevel)
curr_win = cnt
cnt == 0 AND curr_win != NoInterrupt
int_source=curr_win
IntClear
cpu_int_clear[curr_win]=1
Machine remains in same state by default
All outputs are zero unless otherwise stated
State Description:
Idle: Normal idle state
Compare: Compare interrupt level. Determines the interrupt source
IntClear: Interrupt clear, clear the pending bit for the current interrupt vector

FIG. 74

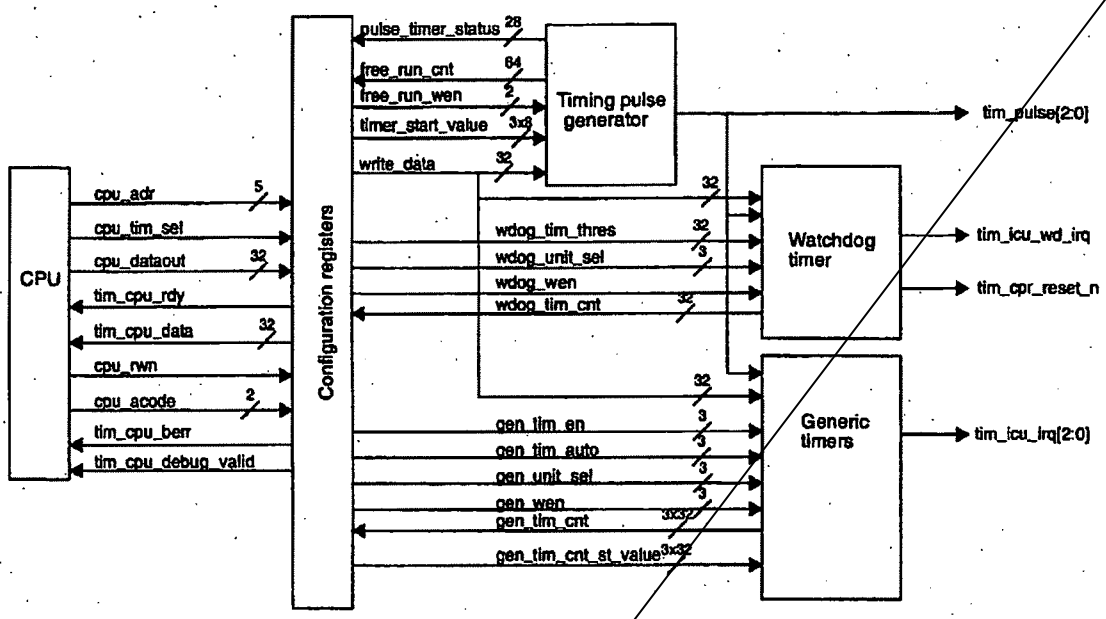


FIG. 75

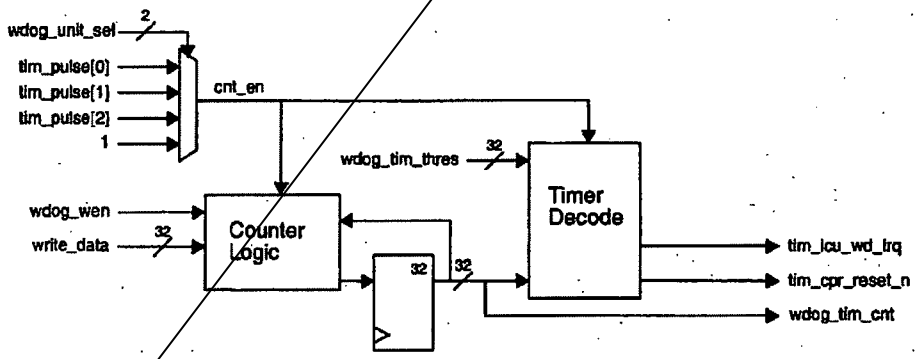


FIG. 76

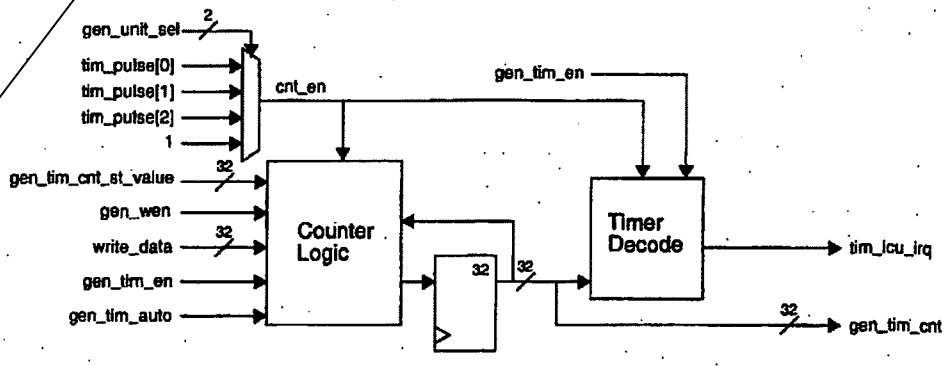


FIG. 77

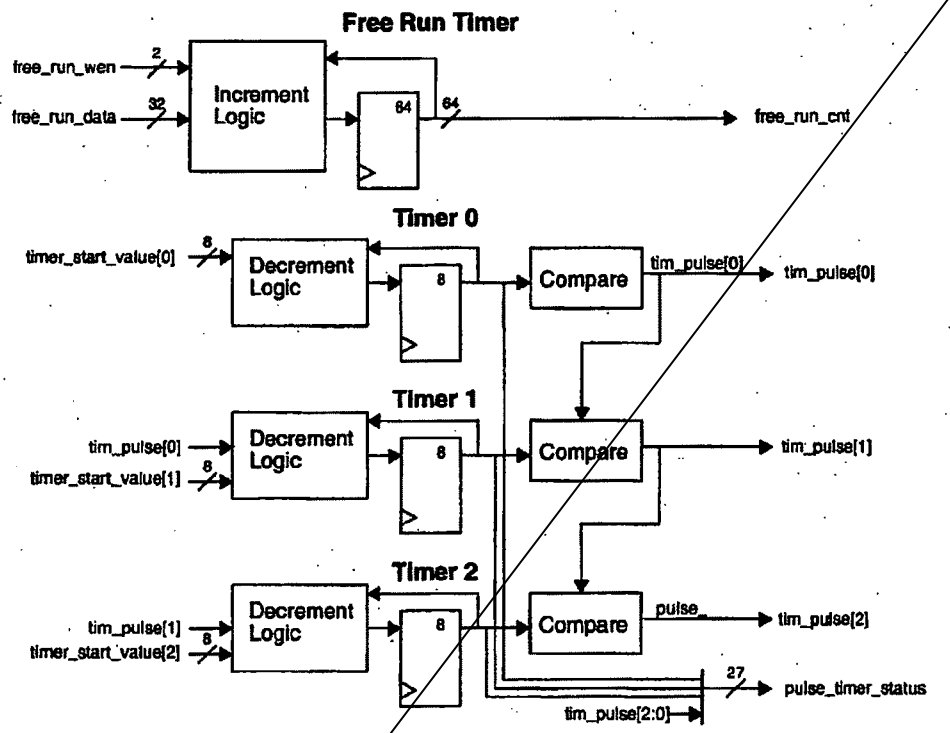

Free Run Timer
free_run_wen
2
free_run_data
32
Increment Logic
64
64
free_run_cnt
Timer 0
timer_start_value[0]
8
Decrement Logic
8
Compare
tim_pulse[0]
tim_pulse[0]
Timer 1
tim_pulse[0]
timer_start_value[1]
8
Decrement Logic
8
Compare
tim_pulse[1]
Timer 2
tim_pulse[1]
timer_start_value[2]
8
Decrement Logic
8
Compare
pulse_
tim_pulse[2]
27
pulse_timer_status
tim_pulse[2:0]

FIG. 78

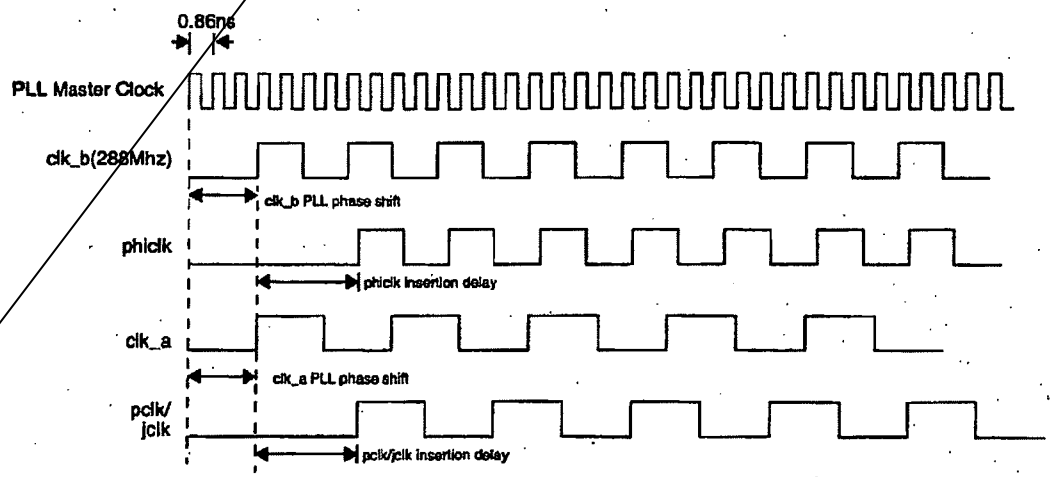

0.86ns
PLL Master Clock
clk_b(288Mhz)
clk_b PLL phase shift
phiclk
phiclk insertion delay
clk_a
clk_a PLL phase shift
pclk/
jclk
pclk/jclk insertion delay

FIG. 79

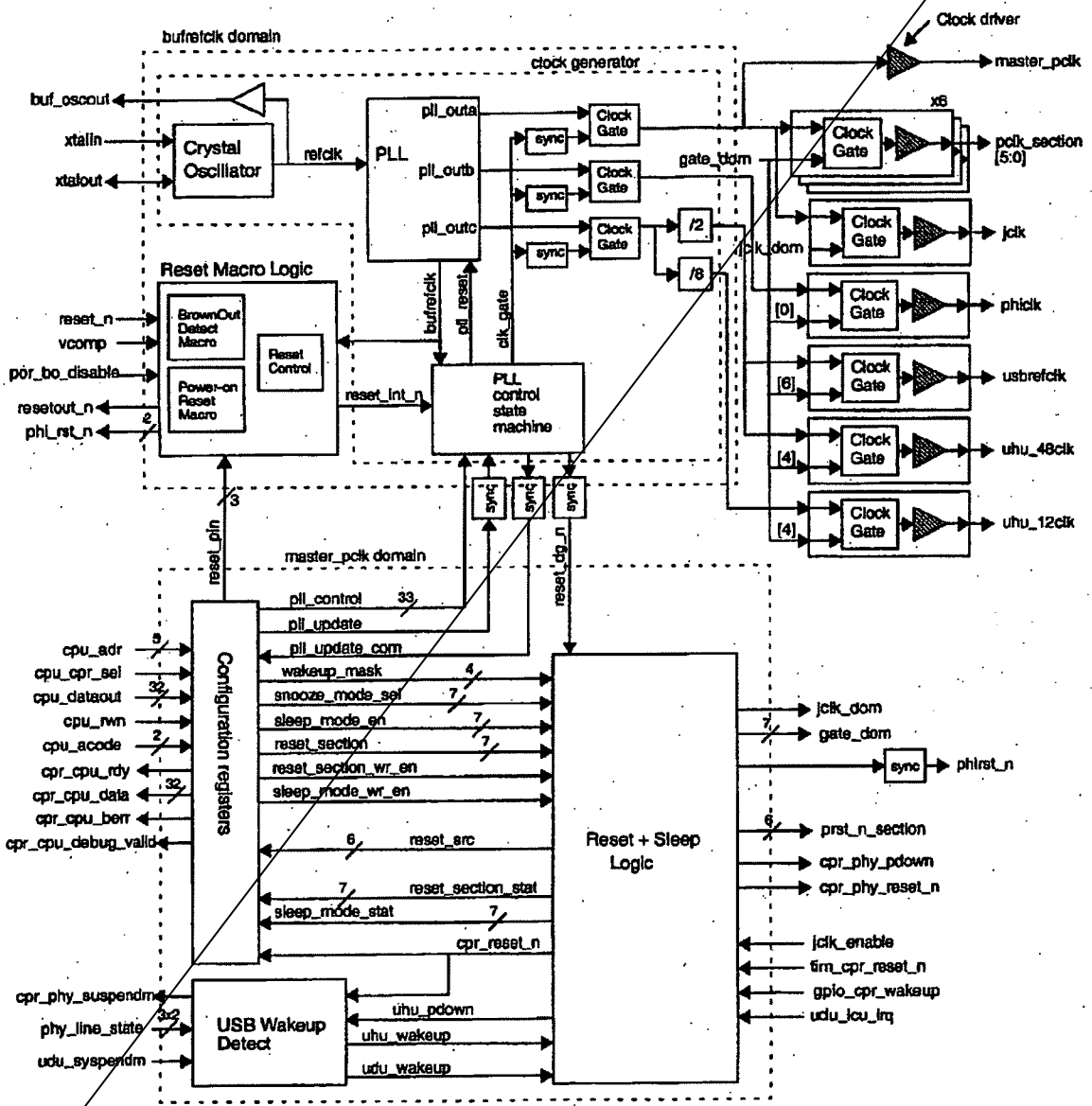

bufrefclk domain
clock generator
Clock driver
master_pclk
buf_oscout
xtalin
xtalout
Crystal Oscillator
refclk
PLL
pll_outa
pll_outb
pll_outc
sync
Clock Gate
gate_dom
x6
pclk_section [5:0]
/2
/8
jclk_dom
jclk
[0]
phiclk
[6]
usbrefclk
[4]
uhu_48clk
[4]
uhu_12clk
Reset Macro Logic
reset_n
vcomp
por_bo_disable
resetout_n
phi_rst_n
BrownOut Detect Macro
Power-on Reset Macro
Reset Control
reset_int_n
bufrefclk
pll_reset
clk_gate
PLL control state machine
reset_pin
master_pclk domain
reset_dg_n
pll_control
pll_update
pll_update_com
wakeup_mask
snooze_mode_sel
sleep_mode_en
reset_section
reset_section_wr_en
sleep_mode_wr_en
reset_src
reset_section_stat
sleep_mode_stat
cpr_reset_n
Configuration registers
cpu_adr
cpu_cpr_sel
cpu_dataout
cpu_rwn
cpu_acode
cpr_cpu_rdy
cpr_cpu_data
cpr_cpu_berr
cpr_cpu_debug_valid
Reset + Sleep Logic
jclk_dom
gate_dom
phirst_n
prst_n_section
cpr_phy_pdown
cpr_phy_reset_n
jclk_enable
tim_cpr_reset_n
gpio_cpr_wakeup
udu_lcu_irq
cpr_phy_suspendm
phy_line_state
udu_sysperidm
USB Wakeup Detect
uhu_pdown
uhu_wakeup
udu_wakeup

FIG. 80

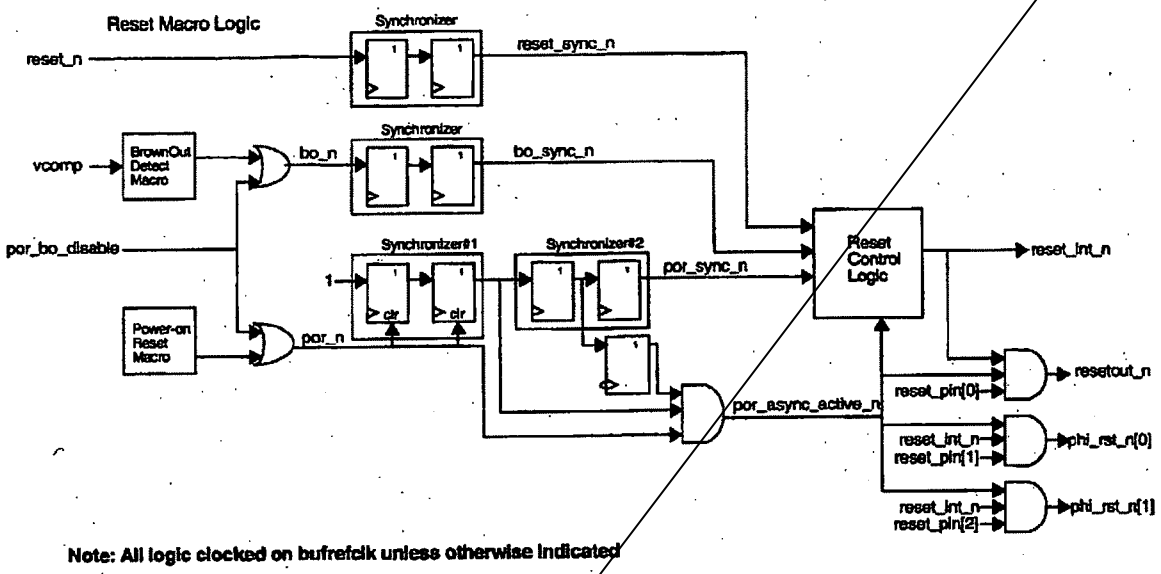

Reset Macro Logic
reset_n
Synchronizer
reset_sync_n
vcomp
BrownOut Detect Macro
bo_n
Synchronizer
bo_sync_n
por_bo_disable
Synchronizer#1
Synchronizer#2
por_sync_n
Reset Control Logic
reset_int_n
Power-on Reset Macro
por_n
por_async_active_n
resetout_n
reset_pin[0]
reset_int_n
reset_pin[1]
phi_rst_n[0]
reset_int_n
reset_pin[2]
phi_rst_n[1]
Note: All logic clocked on bufrefclk unless otherwise indicated

FIG. 81

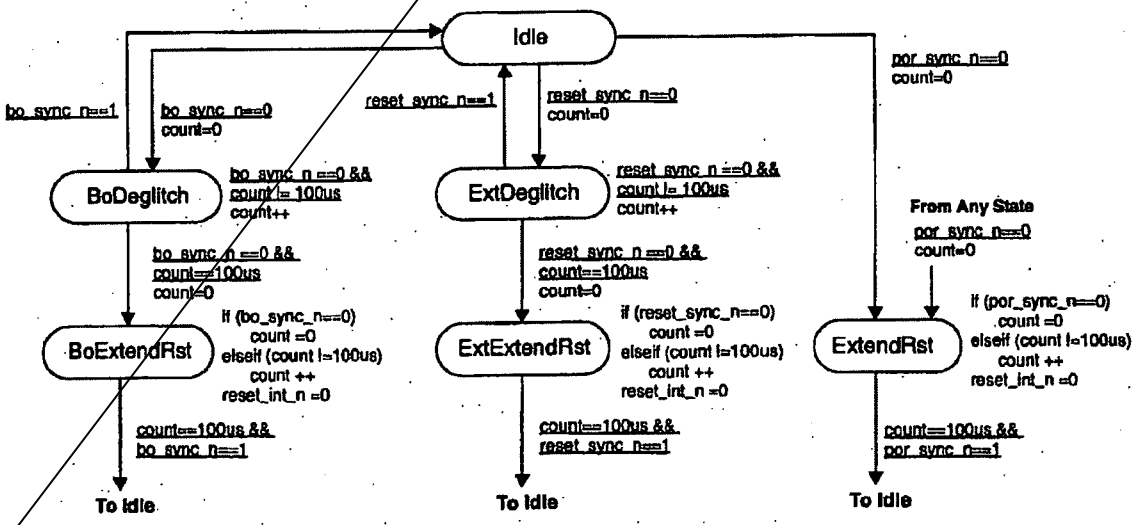

Idle
bo_sync_n==1
bo_sync_n==0
count=0
reset_sync_n==1
reset_sync_n==0
count=0
por_sync_n==0
count=0
BoDeglitch
bo_sync_n ==0 &&
count != 100us
count++
ExtDeglitch
reset_sync_n ==0 &&
count != 100us
count++
From Any State
por_sync_n==0
count=0
bo_sync_n ==0 &&
count==100us
count=0
reset_sync_n ==0 &&
count==100us
count=0
if (bo_sync_n==0)
count =0
elseif (count !=100us)
count ++
reset_int_n =0
if (reset_sync_n==0)
count =0
elseif (count !=100us)
count ++
reset_int_n =0
if (por_sync_n==0)
count =0
elseif (count !=100us)
count ++
reset_int_n =0
BoExtendRst
ExtExtendRst
ExtendRst
count==100us &&
bo_sync_n==1
count==100us &&
reset_sync_n==1
count==100us &&
por_sync_n==1
To Idle
To Idle
To Idle

FIG. 82

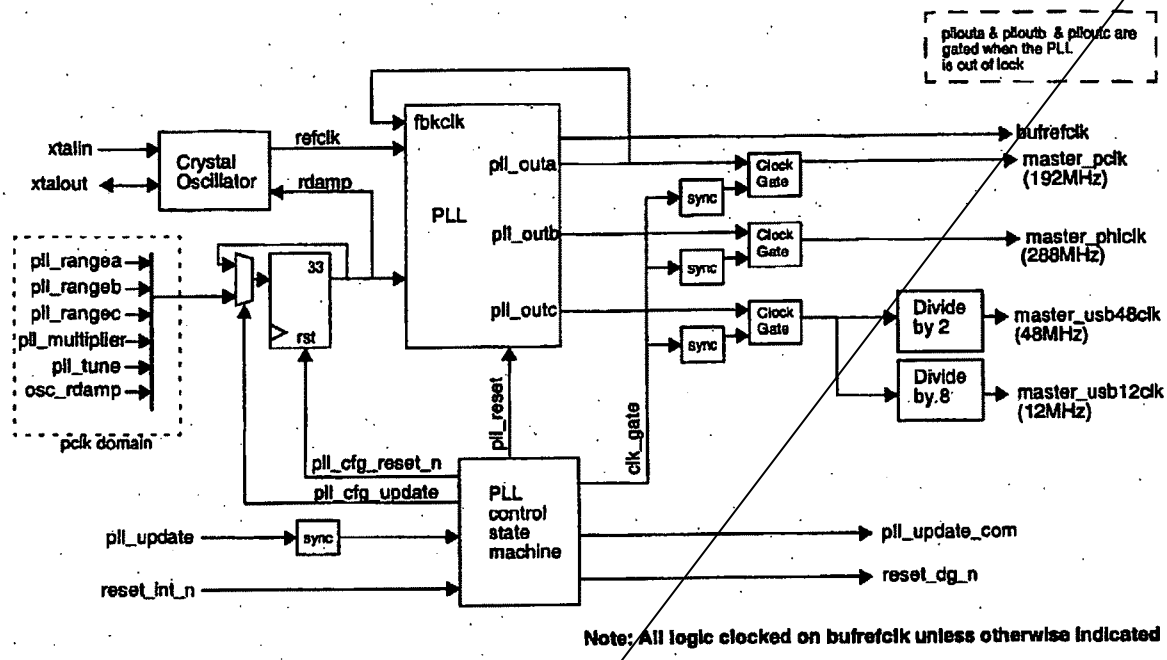

pllouta & plloutb & plloutc are gated when the PLL is out of lock
xtalin
xtalout
Crystal Oscillator
refclk
rdamp
fbkclk
PLL
pll_outa
pll_outb
pll_outc
bufrefclk
master_pclk (192MHz)
master_phiclk (288MHz)
master_usb48clk (48MHz)
master_usb12clk (12MHz)
Clock Gate
sync
Divide by 2
Divide by 8
pll_rangea
pll_rangeb
pll_rangec
pll_multiplier
pll_tune
osc_rdamp
pclk domain
33
rst
pll_reset
clk_gate
pll_cfg_reset_n
pll_cfg_update
PLL control state machine
pll_update
reset_int_n
pll_update_com
reset_dg_n
Note: All logic clocked on bufrefclk unless otherwise indicated

FIG. 83

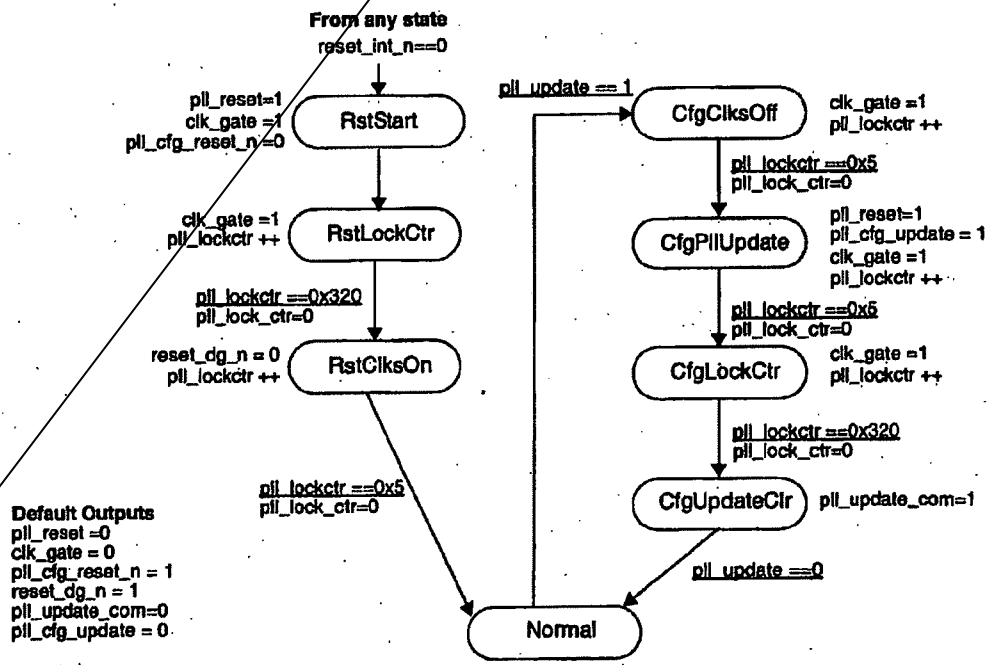

From any state
reset_int_n==0
pll_reset=1
clk_gate =1
pll_cfg_reset_n =0
RstStart
clk_gate =1
pll_lockctr ++
RstLockCtr
pll_lockctr ==0x320
pll_lock_ctr=0
reset_dg_n = 0
pll_lockctr ++
RstClksOn
pll_lockctr ==0x5
pll_lock_ctr=0
Default Outputs
pll_reset =0
clk_gate = 0
pll_cfg_reset_n = 1
reset_dg_n = 1
pll_update_com=0
pll_cfg_update = 0
Normal
pll_update == 1
CfgClksOff
clk_gate =1
pll_lockctr ++
pll_lockctr ==0x5
pll_lock_ctr=0
CfgPllUpdate
pll_reset=1
pll_cfg_update = 1
clk_gate =1
pll_lockctr ++
pll_lockctr ==0x5
pll_lock_ctr=0
CfgLockCtr
clk_gate =1
pll_lockctr ++
pll_lockctr ==0x320
pll_lock_ctr=0
CfgUpdateClr
pll_update_com=1
pll_update ==0

FIG. 84

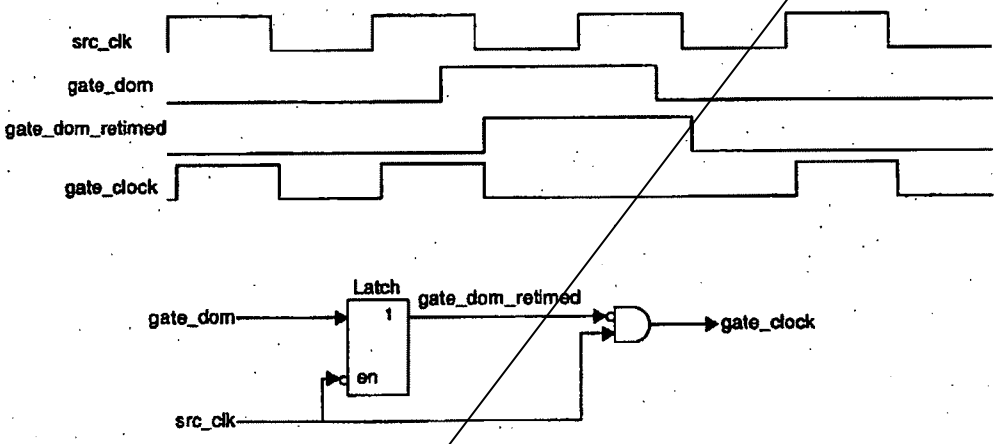

src_clk
gate_dom
gate_dom_retimed
gate_clock
Latch
gate_dom
1
en
gate_dom_retimed
gate_clock
src_clk

FIG. 85

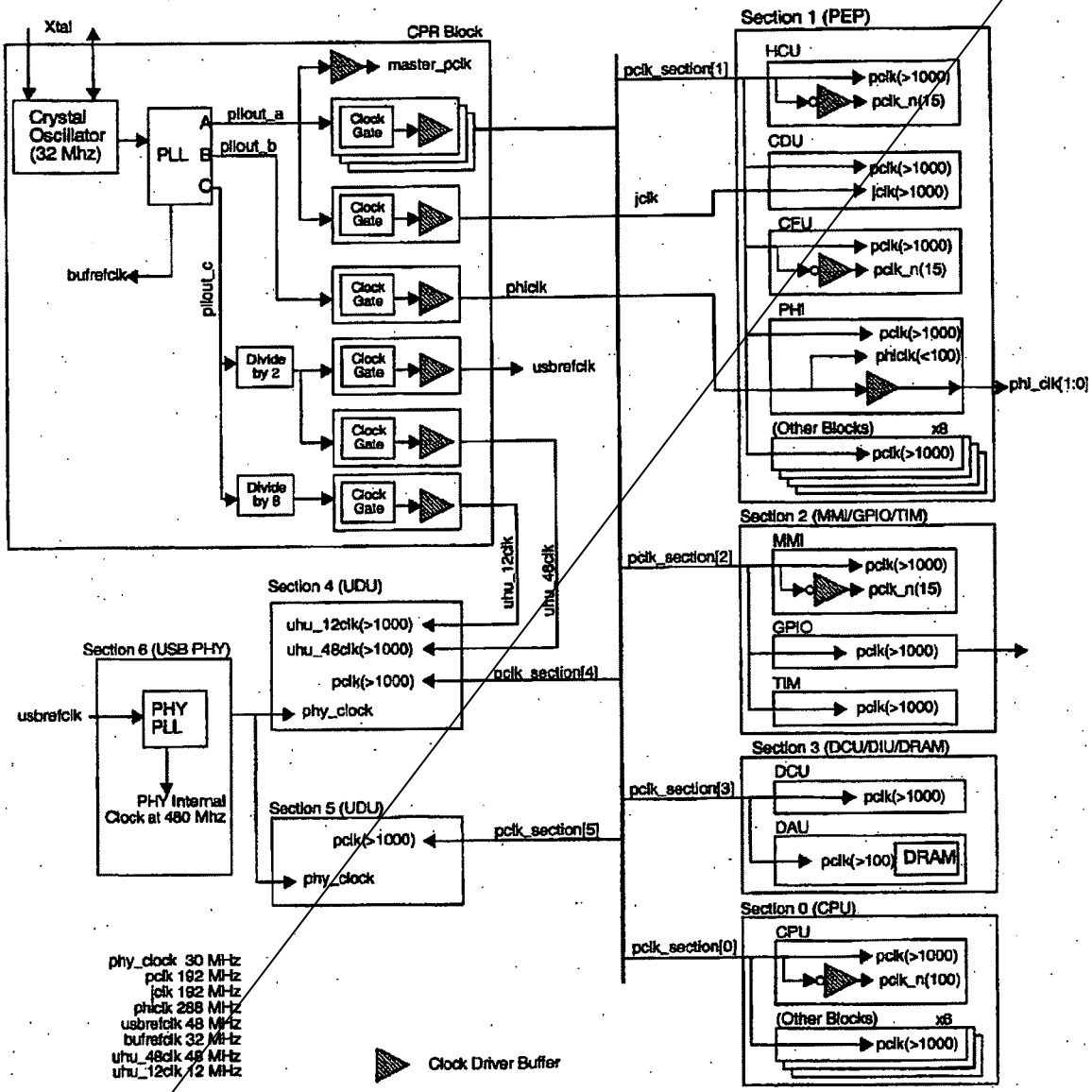

Xtal
CPR Block
Crystal Oscillator (32 Mhz)
PLL
A
B
C
pllout_a
pllout_b
pllout_c
bufrefclk
master_pclk
Clock Gate
Divide by 2
Divide by 8
phiclk
usbrefclk
uhu_12clk
uhu_48clk
pclk_section[1]
jclk
pclk_section[2]
pclk_section[4]
pclk_section[3]
pclk_section[5]
pclk_section[0]
Section 1 (PEP)
HCU
CDU
CFU
PHI
(Other Blocks)
x8
pclk(>1000)
pclk_n(15)
jclk(>1000)
phiclk(<100)
phi_clk[1:0]
Section 2 (MMI/GPIO/TIM)
MMI
GPIO
TIM
Section 3 (DCU/DIU/DRAM)
DCU
DAU
DRAM
pclk(>100)
Section 0 (CPU)
CPU
pclk_n(100)
Section 4 (UDU)
uhu_12clk(>1000)
uhu_48clk(>1000)
phy_clock
Section 5 (UDU)
Section 6 (USB PHY)
PHY PLL
PHY Internal Clock at 480 Mhz
phy_clock 30 MHz
pclk 192 MHz
jclk 192 MHz
phiclk 288 MHz
usbrefclk 48 MHz
bufrefclk 32 MHz
uhu_48clk 48 MHz
uhu_12clk 12 MHz
Clock Driver Buffer

FIG. 86

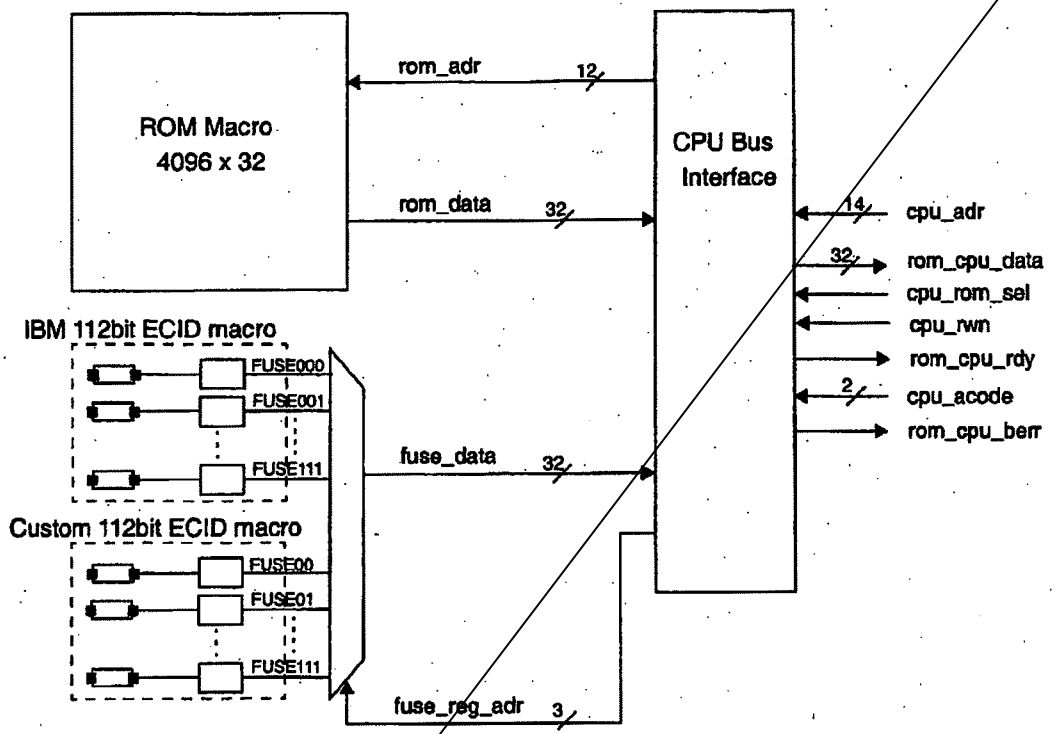

ROM Macro
4096 x 32
rom_adr
12
rom_data
32
CPU Bus
Interface
14
cpu_adr
32
rom_cpu_data
cpu_rom_sel
cpu_rwn
rom_cpu_rdy
2
cpu_acode
rom_cpu_berr
IBM 112bit ECID macro
FUSE000
FUSE001
FUSE111
fuse_data
32
Custom 112bit ECID macro
FUSE00
FUSE01
FUSE111
fuse_reg_adr
3

FIG. 87

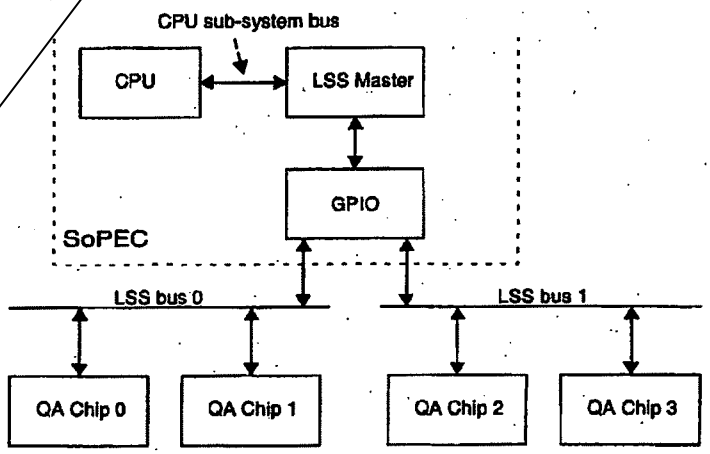

CPU sub-system bus
CPU
LSS Master
GPIO
SoPEC
LSS bus 0
LSS bus 1
QA Chip 0
QA Chip 1
QA Chip 2
QA Chip 3

FIG. 88

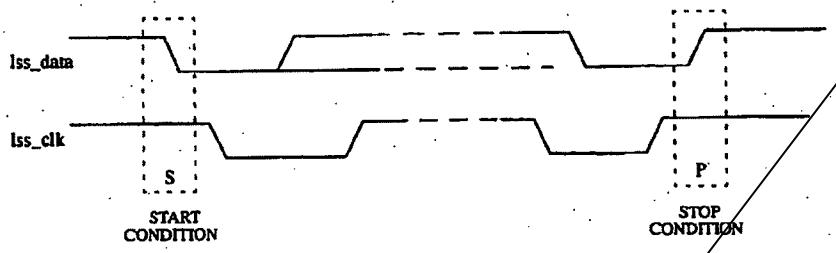


FIG. 89

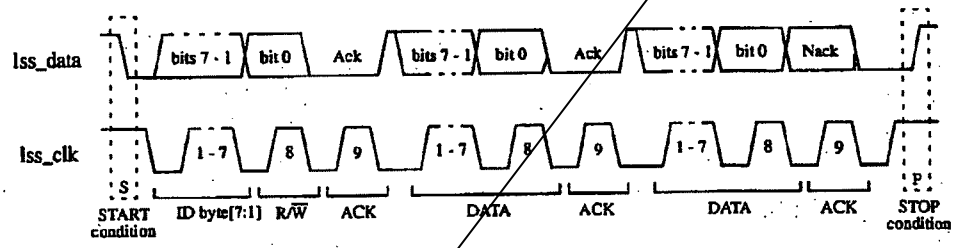


FIG. 90

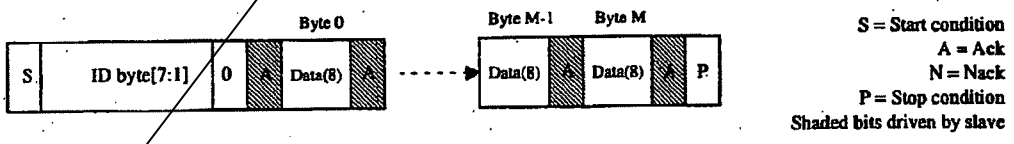


FIG. 91

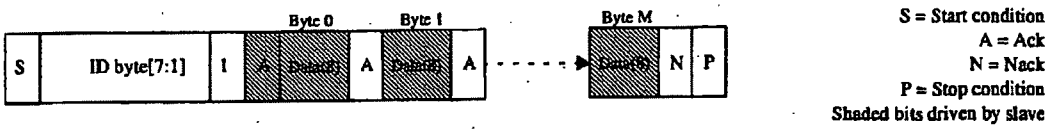


FIG. 92

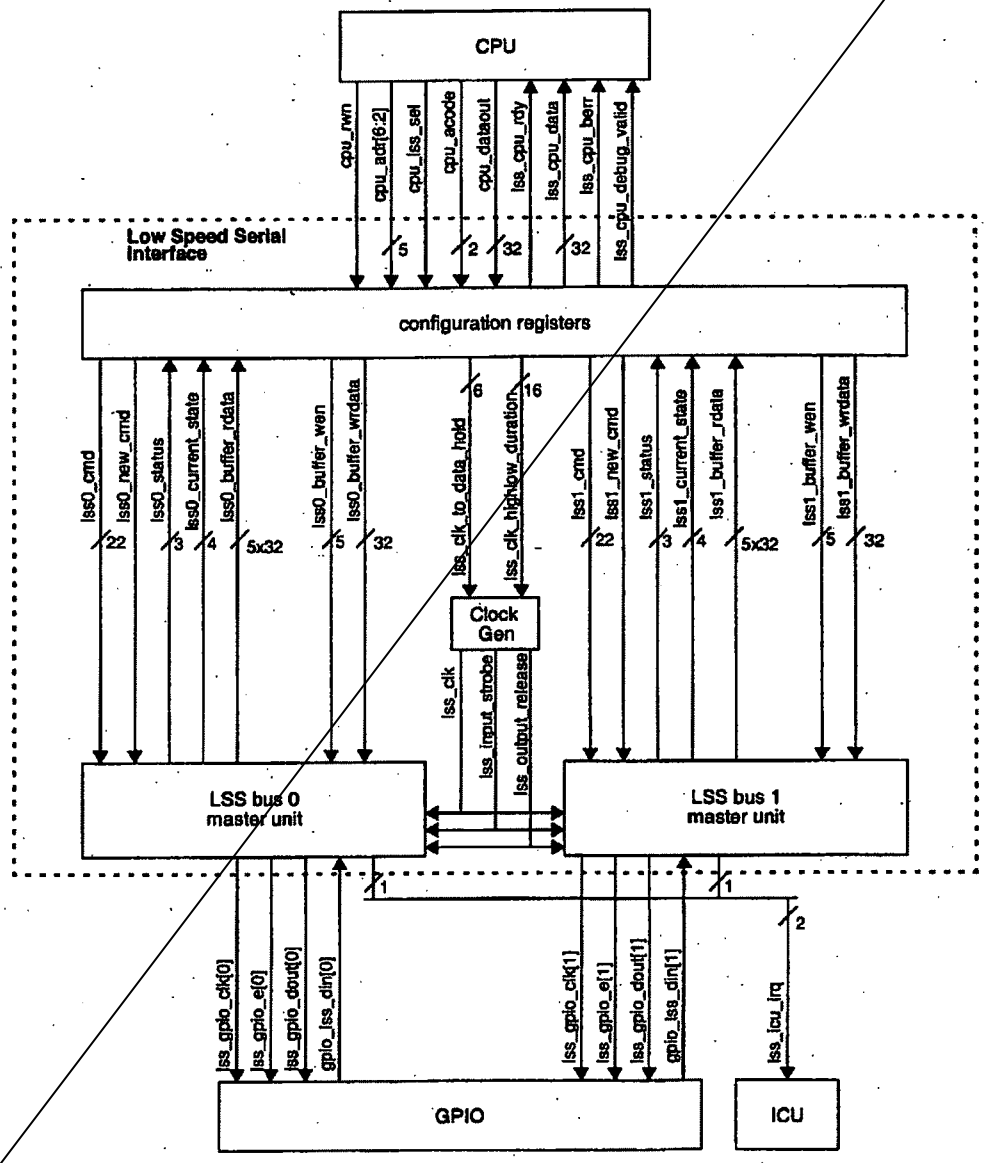

CPU
cpu_rwn
cpu_adr[6:2]
cpu_lss_sel
cpu_acode
cpu_dataout
lss_cpu_rdy
lss_cpu_data
lss_cpu_berr
lss_cpu_debug_valid
5
2
32
32
Low Speed Serial Interface
configuration registers
lss0_cmd
lss0_new_cmd
lss0_status
lss0_current_state
lss0_buffer_rdata
lss0_buffer_wen
lss0_buffer_wrdata
lss_clk_to_data_hold
lss_clk_high/low_duration
6
16
22
3
4
5x32
5
32
lss1_cmd
lss1_new_cmd
lss1_status
lss1_current_state
lss1_buffer_rdata
lss1_buffer_wen
lss1_buffer_wrdata
22
3
4
5x32
5
32
Clock Gen
lss_clk
lss_input_strobe
lss_output_release
LSS bus 0 master unit
LSS bus 1 master unit
1
1
2
lss_gpio_clk[0]
lss_gpio_e[0]
lss_gpio_dout[0]
gpio_lss_din[0]
lss_gpio_clk[1]
lss_gpio_e[1]
lss_gpio_dout[1]
gpio_lss_din[1]
lss_icu_irq
GPIO
ICU

FIG. 93

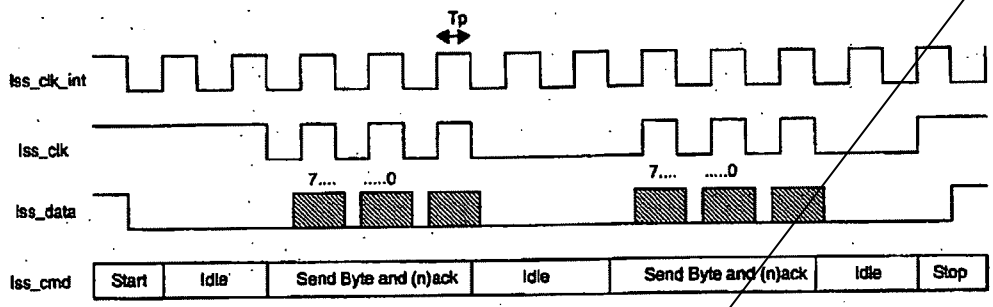

Tp
Iss_clk_int
Iss_clk
7.... 0
Iss_data
7.... 0
Iss_cmd
Start
Idle
Send Byte and (n)ack
Idle
Send Byte and (n)ack
Idle
Stop

FIG. 94

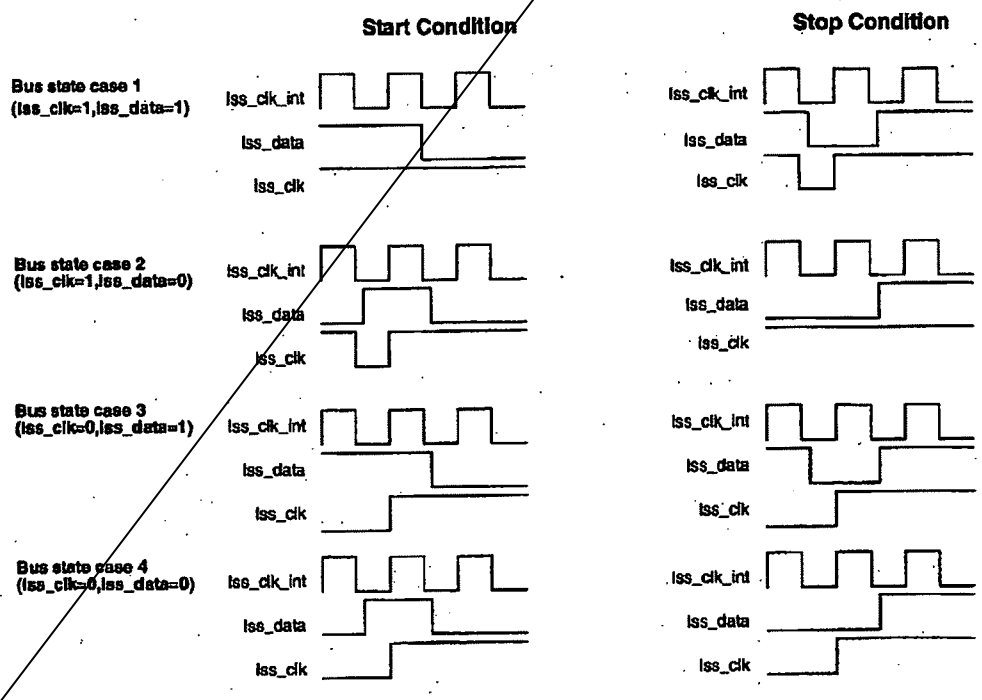

Start Condition
Stop Condition
Bus state case 1
(Iss_clk=1,Iss_data=1)
Iss_clk_int
Iss_data
Iss_clk
Bus state case 2
(Iss_clk=1,Iss_data=0)
Iss_clk_int
Iss_data
Iss_clk
Bus state case 3
(Iss_clk=0,Iss_data=1)
Iss_clk_int
Iss_data
Iss_clk
Bus state case 4
(Iss_clk=0,Iss_data=0)
Iss_clk_int
Iss_data
Iss_clk

FIG. 95

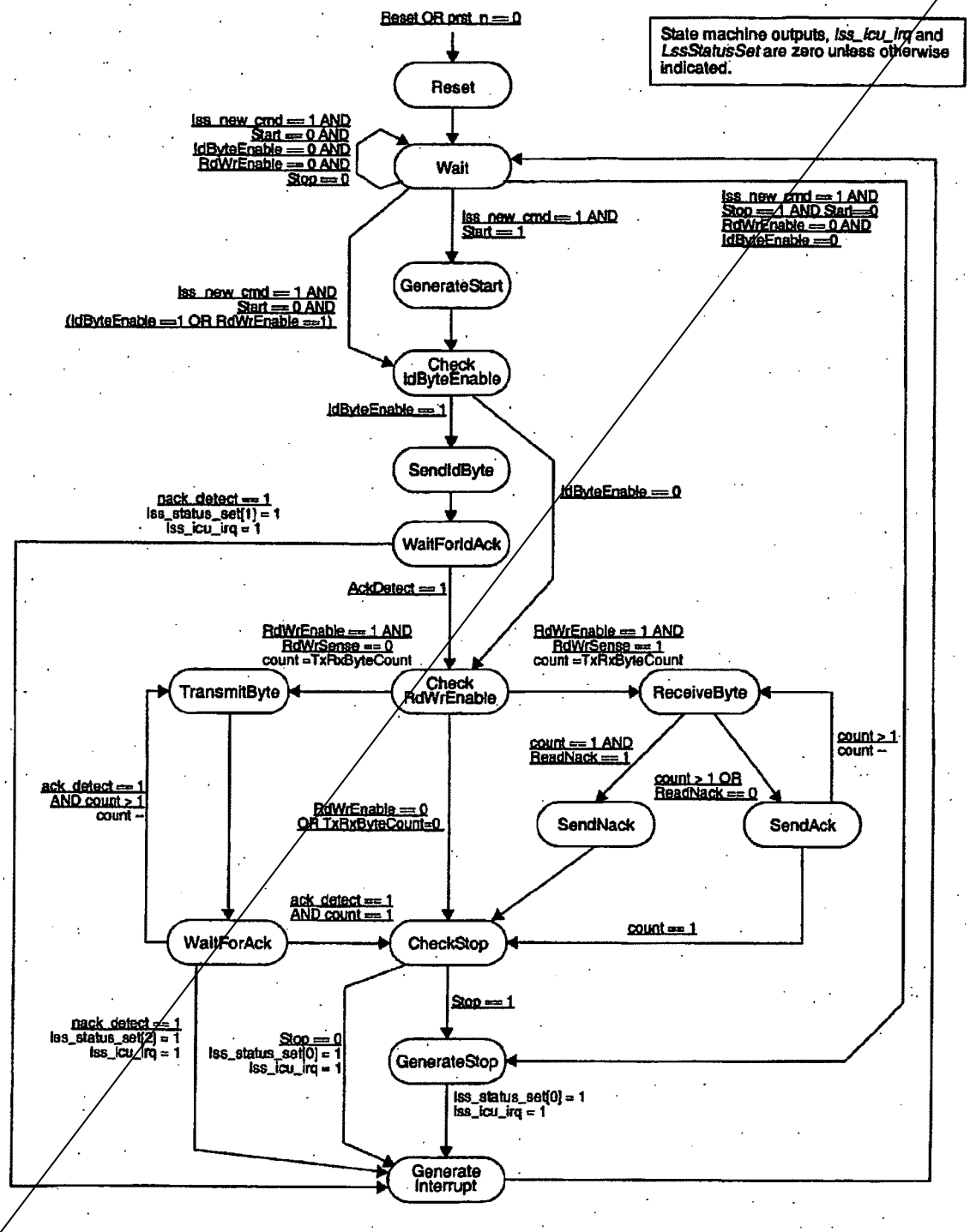

State machine outputs, lss_icu_irq and LssStatusSet are zero unless otherwise indicated.
Reset OR prst_n == 0
Reset
lss_new_cmd == 1 AND Start == 0 AND IdByteEnable == 0 AND RdWrEnable == 0 AND Stop == 0
Wait
lss_new_cmd == 1 AND Start == 1
lss_new_cmd == 1 AND Stop == 1 AND Start == 0 RdWrEnable == 0 AND IdByteEnable == 0
GenerateStart
lss_new_cmd == 1 AND Start == 0 AND (IdByteEnable == 1 OR RdWrEnable == 1)
Check IdByteEnable
IdByteEnable == 1
SendIdByte
IdByteEnable == 0
nack_detect == 1 lss_status_set[1] = 1 lss_icu_irq = 1
WaitForIdAck
AckDetect == 1
RdWrEnable == 1 AND RdWrSense == 0 count = TxRxByteCount
RdWrEnable == 1 AND RdWrSense == 1 count = TxRxByteCount
TransmitByte
Check RdWrEnable
ReceiveByte
count == 1 AND ReadNack == 1
count > 1 count --
count > 1 OR ReadNack == 0
ack_detect == 1 AND count > 1 count --
RdWrEnable == 0 OR TxRxByteCount=0
SendNack
SendAck
ack_detect == 1 AND count == 1
count == 1
WaitForAck
CheckStop
Stop == 1
nack_detect == 1 lss_status_set[2] = 1 lss_icu_irq = 1
Stop == 0 lss_status_set[0] = 1 lss_icu_irq = 1
GenerateStop
lss_status_set[0] = 1 lss_icu_irq = 1
Generate Interrupt

FIG. 96

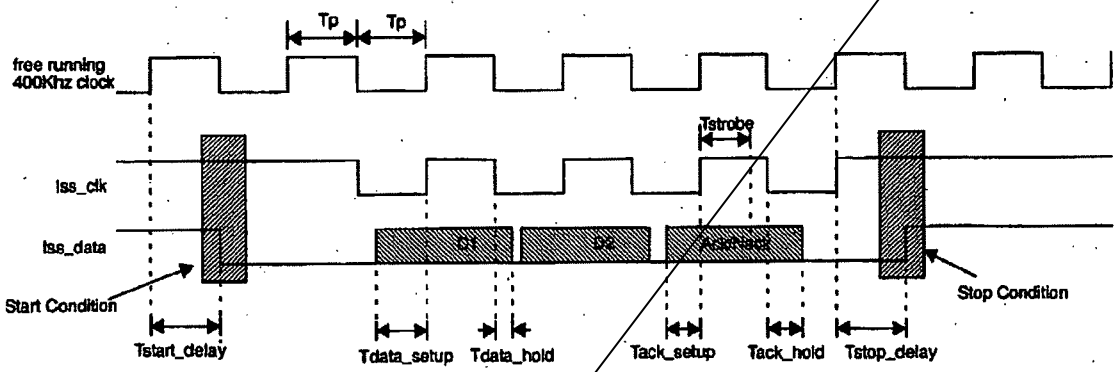

Tp
Tp
free running
400Khz clock
Tstrobe
Iss_clk
Iss_data
D1
D2
Acknowledge
Start Condition
Stop Condition
Tstart_delay
Tdata_setup
Tdata_hold
Tack_setup
Tack_hold
Tstop_delay

FIG. 97

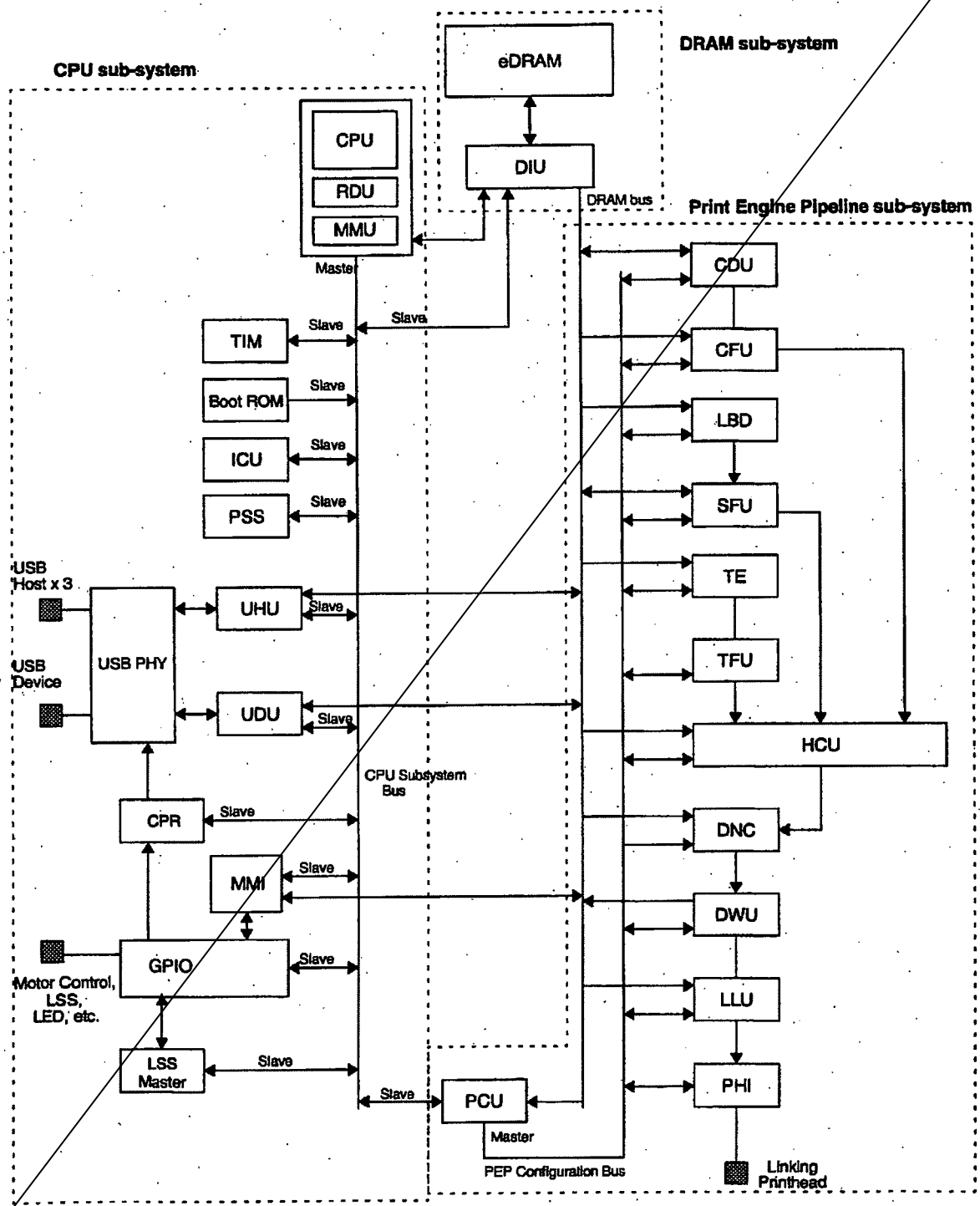

CPU sub-system
DRAM sub-system
eDRAM
CPU
RDU
MMU
DIU
DRAM bus
Print Engine Pipeline sub-system
Master
Slave
TIM
Boot ROM
ICU
PSS
USB Host x 3
USB Device
USB PHY
UHU
UDU
CPU Subsystem Bus
CPR
MMI
GPIO
Motor Control, LSS, LED, etc.
LSS Master
PCU
PEP Configuration Bus
CDU
CFU
LBD
SFU
TE
TFU
HCU
DNC
DWU
LLU
PHI
Linking Printhead

FIG. 98

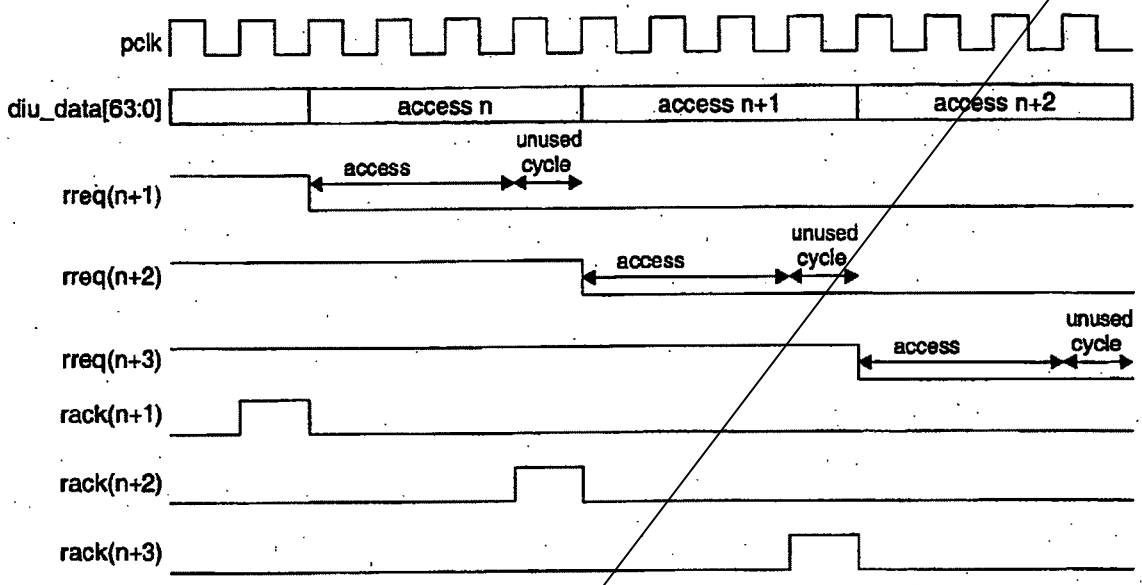


FIG. 99

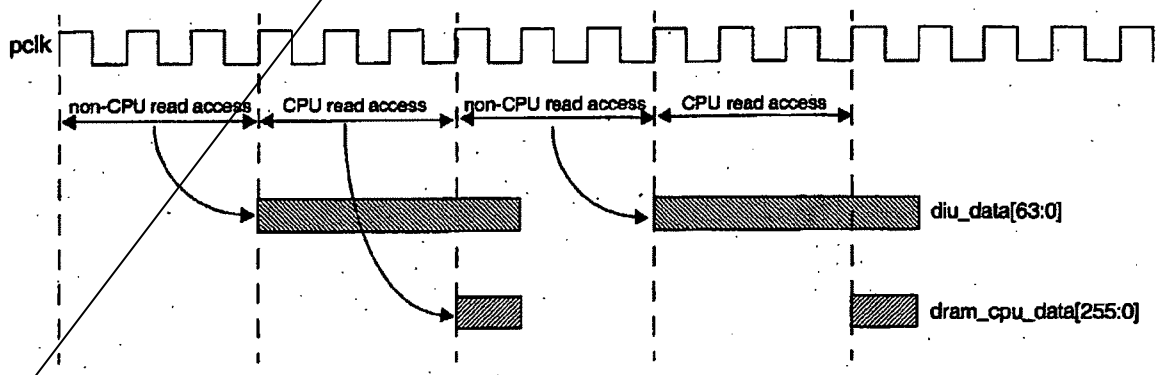


FIG. 100

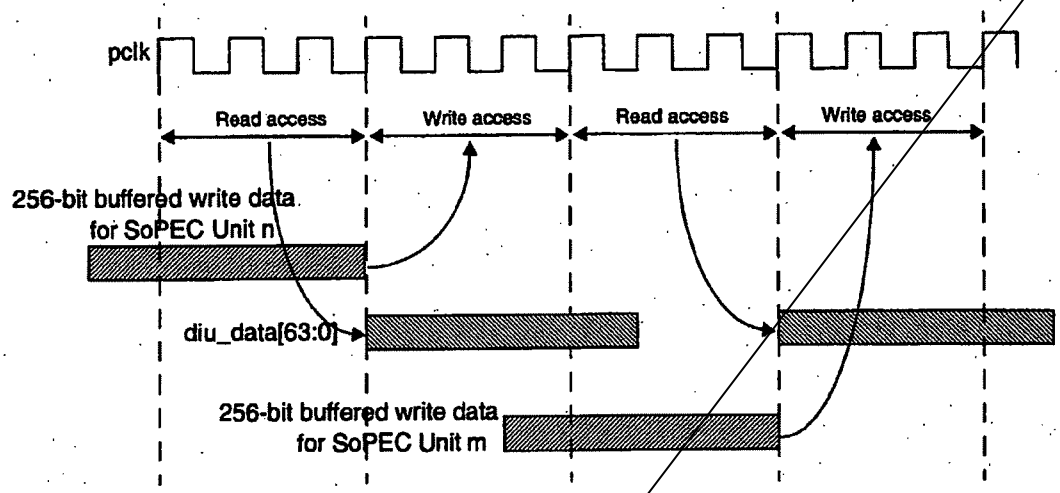


FIG. 101

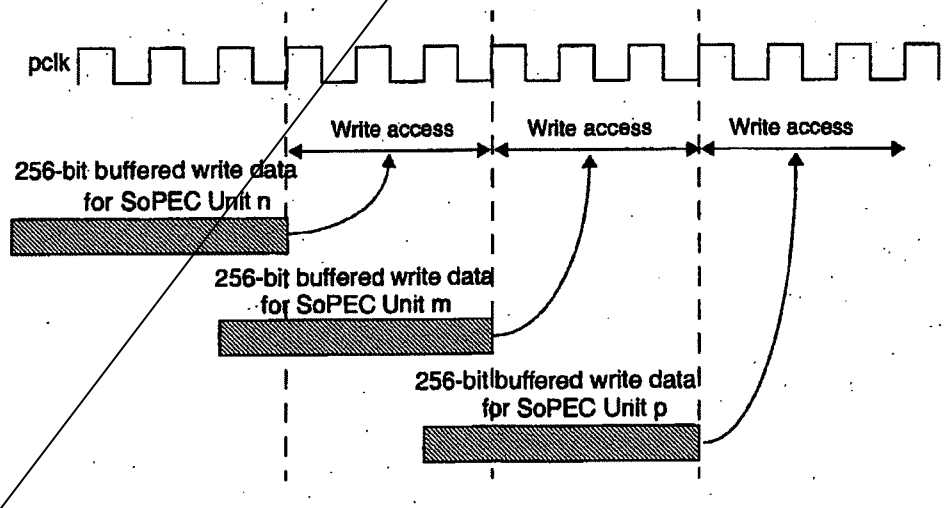


FIG. 102

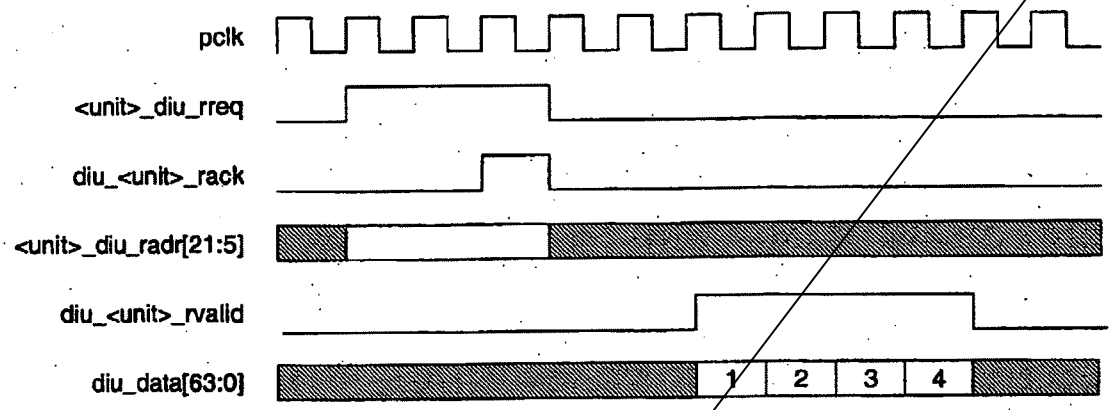


FIG. 103

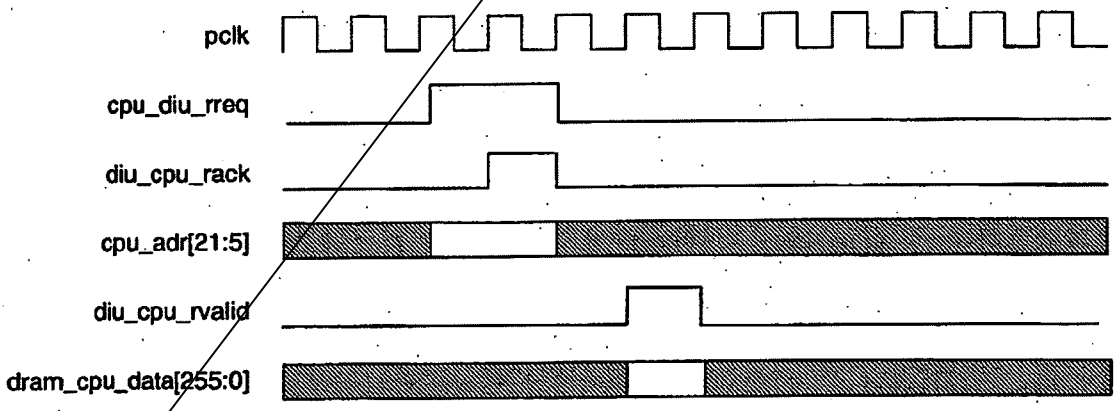


FIG. 104

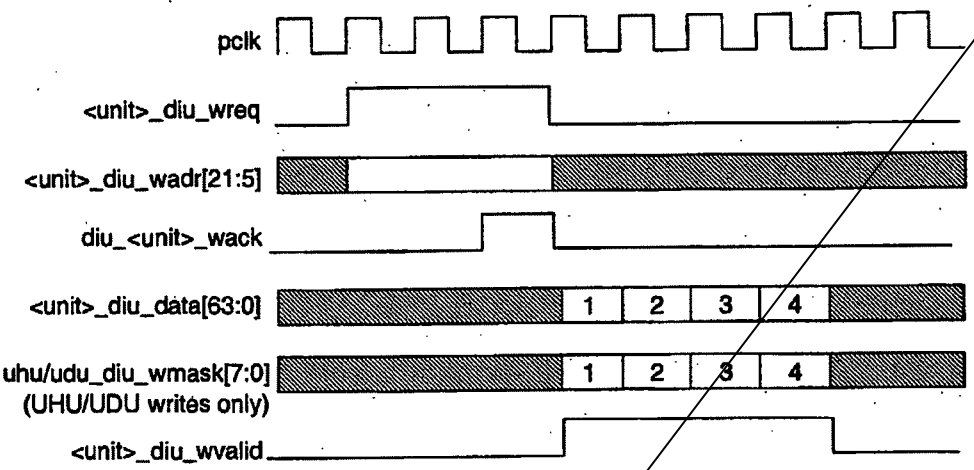


FIG. 105

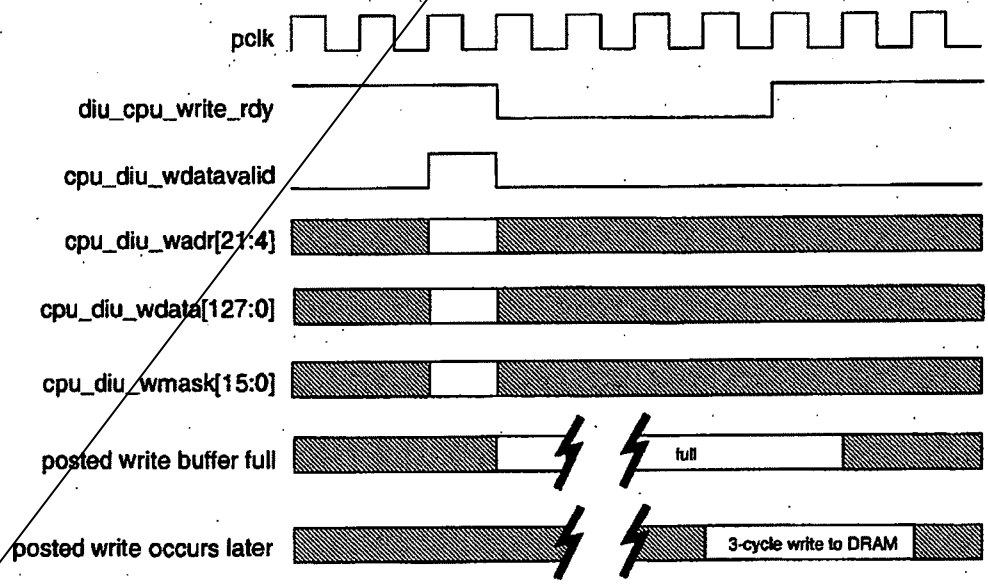


FIG. 106

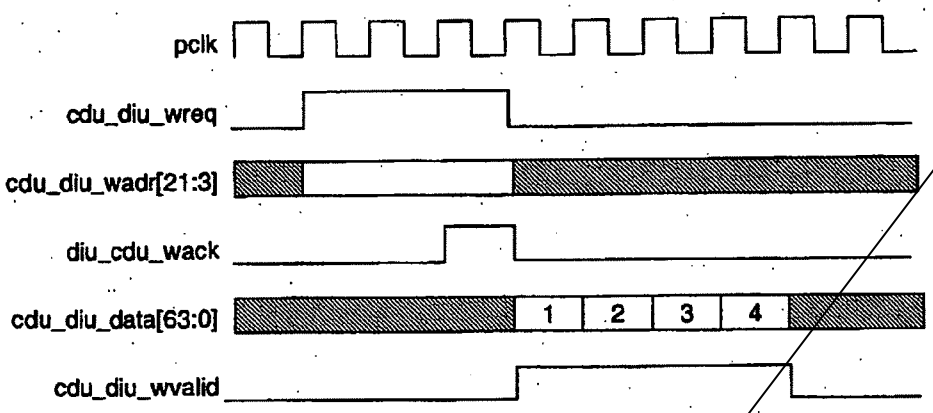


FIG. 107

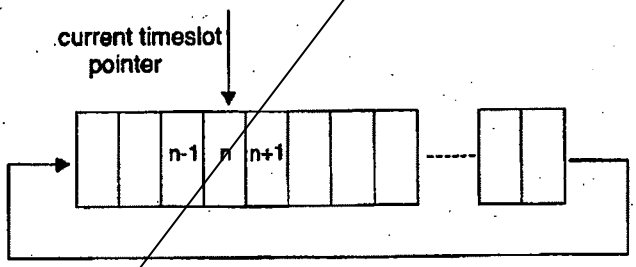


FIG. 108

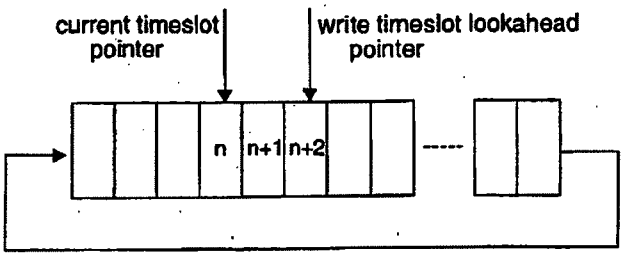


FIG. 109

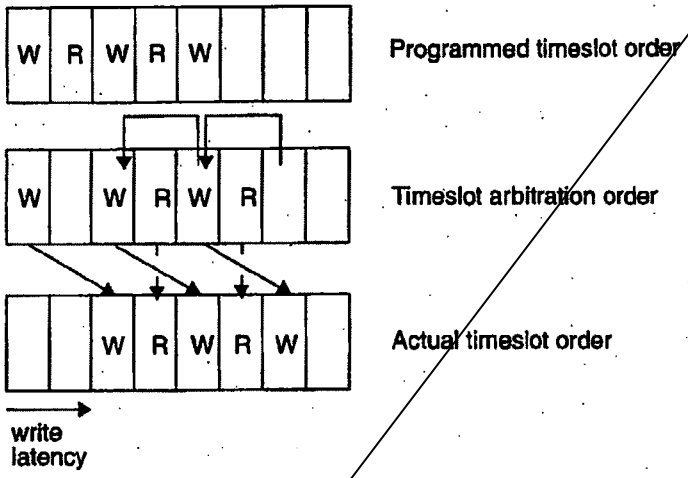


FIG. 110

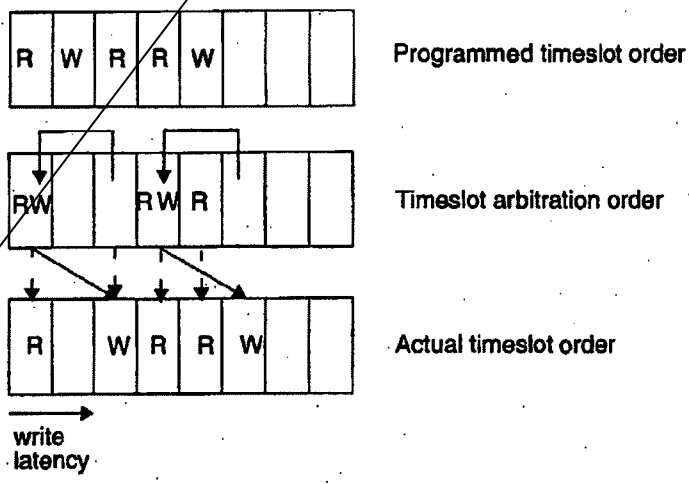


FIG. 111

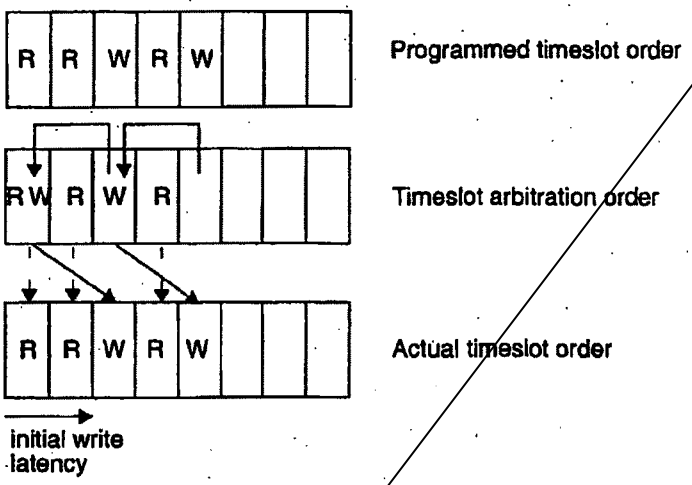


FIG. 112

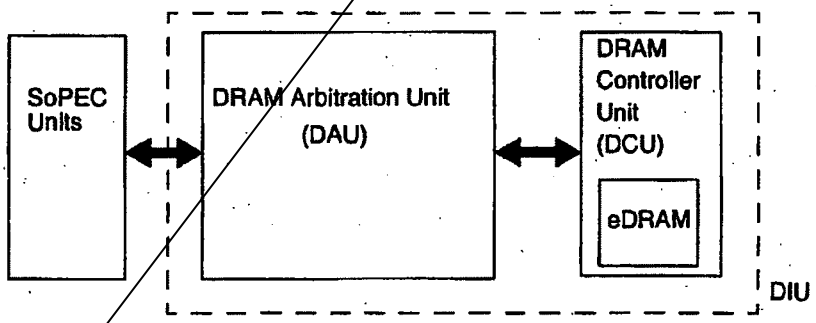


FIG. 113

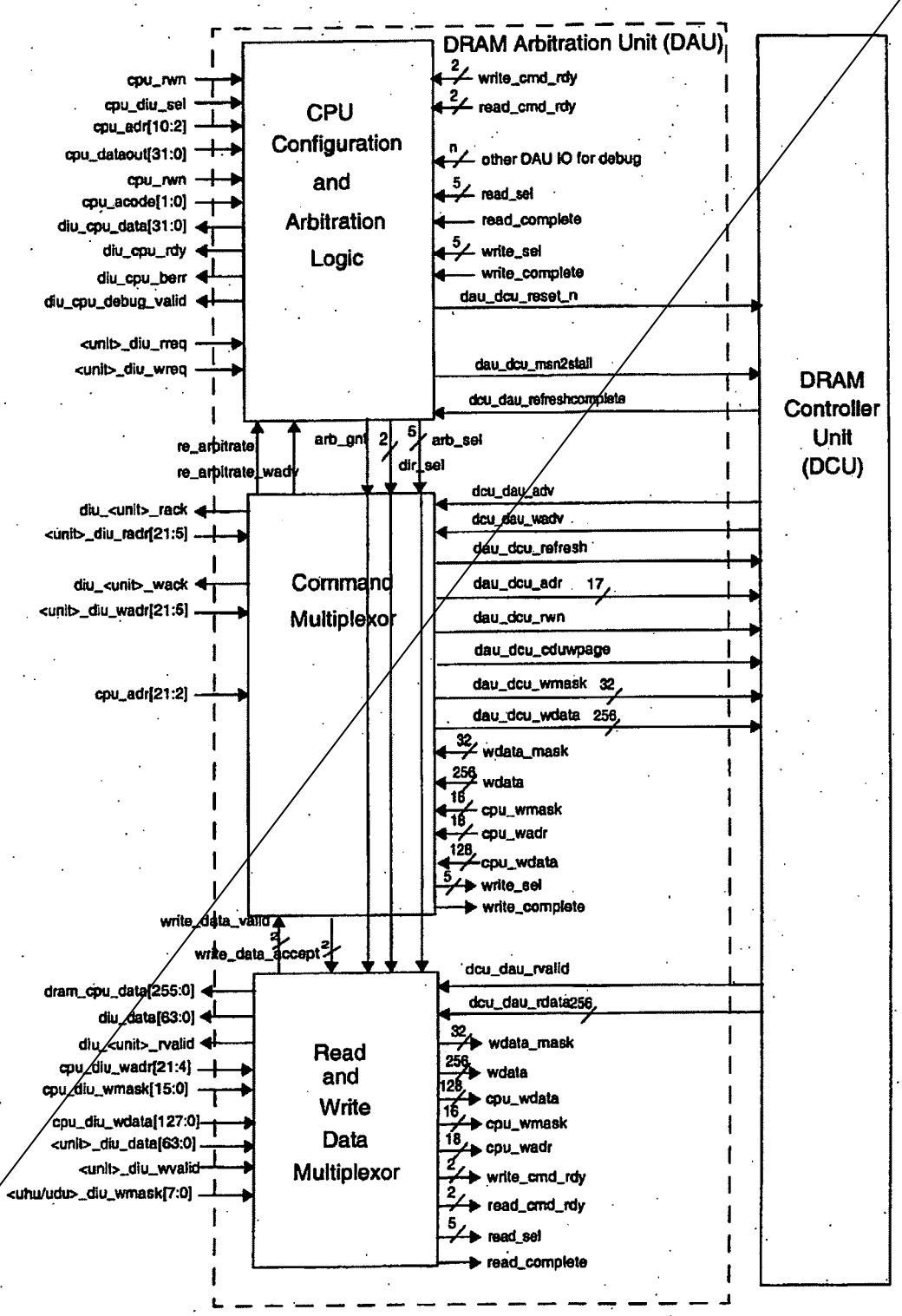

DRAM Arbitration Unit (DAU)
CPU Configuration and Arbitration Logic
Command Multiplexor
Read and Write Data Multiplexor
DRAM Controller Unit (DCU)
cpu_rwn
cpu_diu_sel
cpu_adr[10:2]
cpu_dataout[31:0]
cpu_rwn
cpu_acode[1:0]
diu_cpu_data[31:0]
diu_cpu_rdy
diu_cpu_berr
diu_cpu_debug_valid
<unit>_diu_rreq
<unit>_diu_wreq
re_arbitrate
re_arbitrate_wadv
diu_<unit>_rack
<unit>_diu_radr[21:5]
diu_<unit>_wack
<unit>_diu_wadr[21:5]
cpu_adr[21:2]
write_data_valid
write_data_accept
dram_cpu_data[255:0]
diu_data[63:0]
diu_<unit>_rvalid
cpu_diu_wadr[21:4]
cpu_diu_wmask[15:0]
cpu_diu_wdata[127:0]
<unit>_diu_data[63:0]
<unit>_diu_wvalid
<uhu/udu>_diu_wmask[7:0]
write_cmd_rdy
read_cmd_rdy
other DAU IO for debug
read_sel
read_complete
write_sel
write_complete
dau_dcu_reset_n
dau_dcu_msn2stall
dcu_dau_refreshcomplete
arb_gnt
arb_sel
dir_sel
dcu_dau_adv
dcu_dau_wadv
dau_dcu_refresh
dau_dcu_adr
dau_dcu_rwn
dau_dcu_cduwpage
dau_dcu_wmask
dau_dcu_wdata
wdata_mask
wdata
cpu_wmask
cpu_wadr
cpu_wdata
write_sel
write_complete
dcu_dau_rvalid
dcu_dau_rdata256
wdata_mask
wdata
cpu_wdata
cpu_wmask
cpu_wadr
write_cmd_rdy
read_cmd_rdy
read_sel
read_complete

FIG. 114

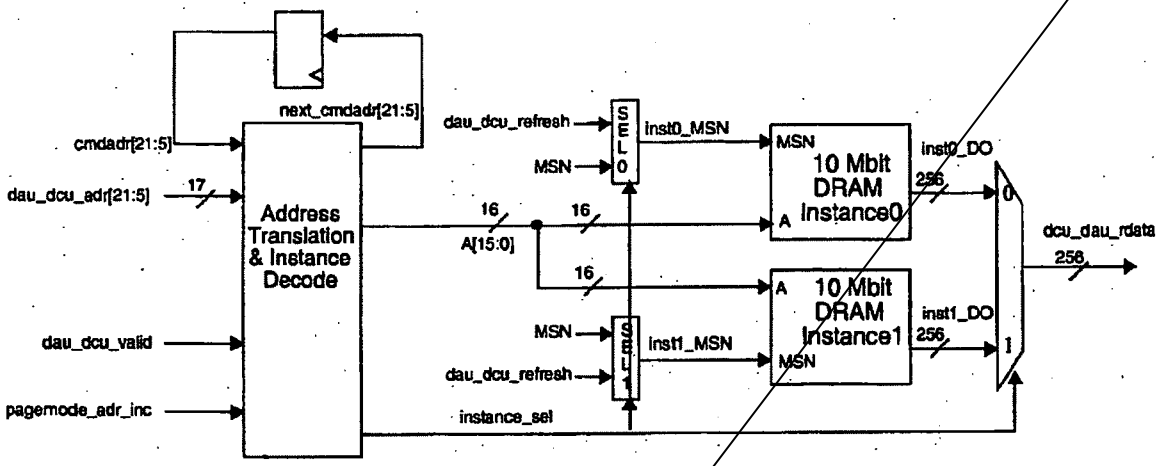


FIG. 115

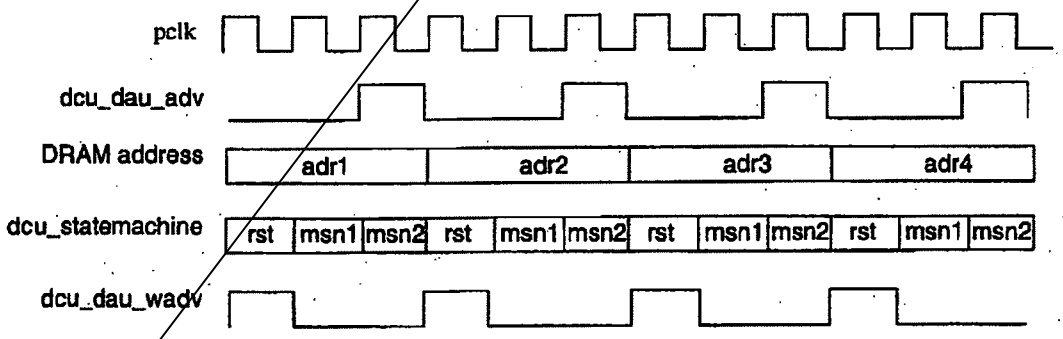


FIG. 116

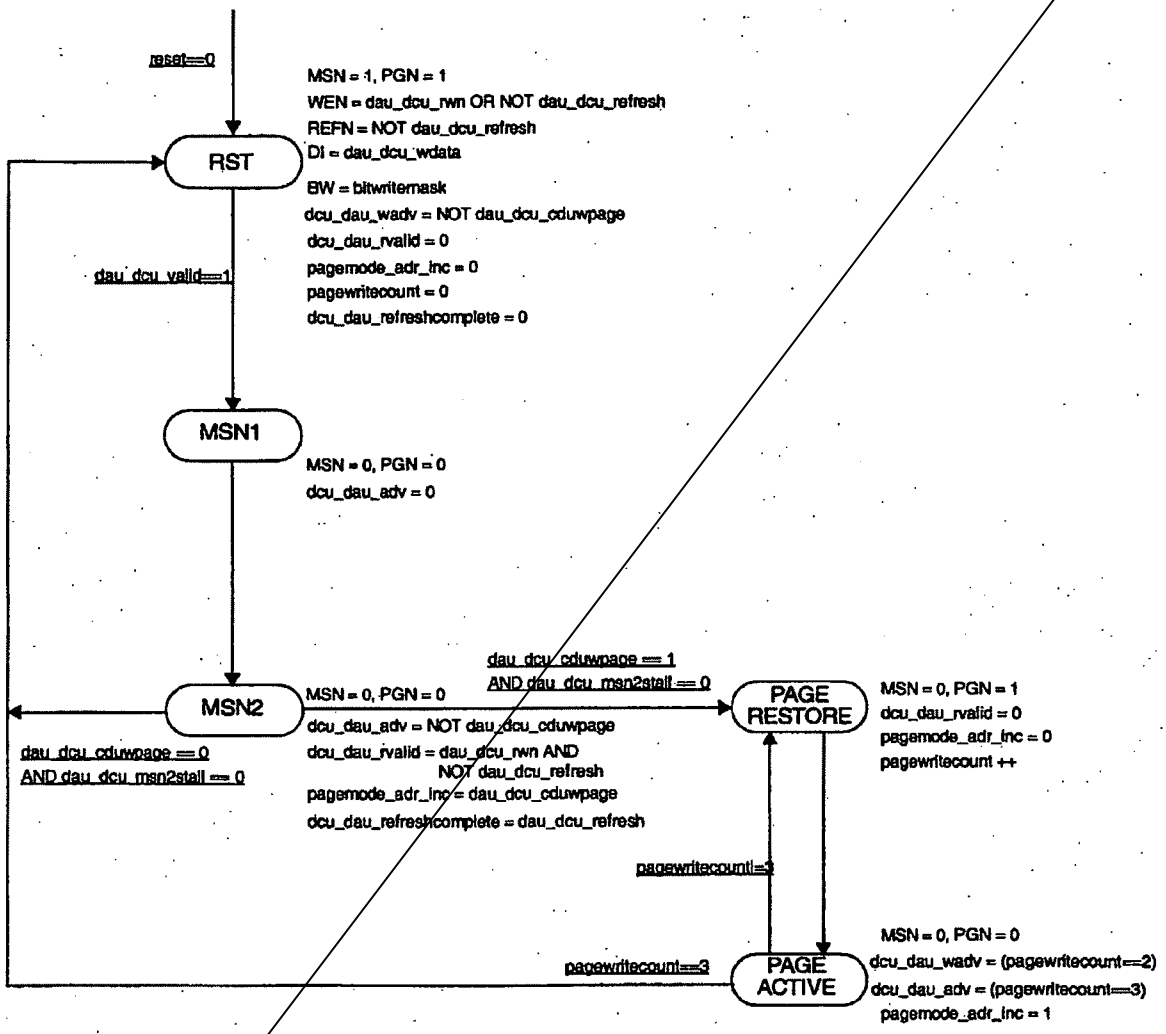

reset==0
MSN = 1, PGN = 1
WEN = dau_dcu_rwn OR NOT dau_dcu_refresh
REFN = NOT dau_dcu_refresh
DI = dau_dcu_wdata
RST
BW = bitwritemask
dcu_dau_wadv = NOT dau_dcu_cduwpage
dcu_dau_rvalid = 0
dau_dcu_valid==1
pagemode_adr_inc = 0
pagewritecount = 0
dcu_dau_refreshcomplete = 0
MSN1
MSN = 0, PGN = 0
dcu_dau_adv = 0
dau_dcu_cduwpage == 1
AND dau_dcu_msn2stall == 0
MSN2
MSN = 0, PGN = 0
dcu_dau_adv = NOT dau_dcu_cduwpage
dcu_dau_rvalid = dau_dcu_rwn AND
NOT dau_dcu_refresh
pagemode_adr_inc = dau_dcu_cduwpage
dcu_dau_refreshcomplete = dau_dcu_refresh
dau_dcu_cduwpage == 0
AND dau_dcu_msn2stall == 0
PAGE RESTORE
MSN = 0, PGN = 1
dcu_dau_rvalid = 0
pagemode_adr_inc = 0
pagewritecount ++
pagewritecount!=3
pagewritecount==3
PAGE ACTIVE
MSN = 0, PGN = 0
dcu_dau_wadv = (pagewritecount==2)
dcu_dau_adv = (pagewritecount==3)
pagemode_adr_inc = 1

FIG. 117

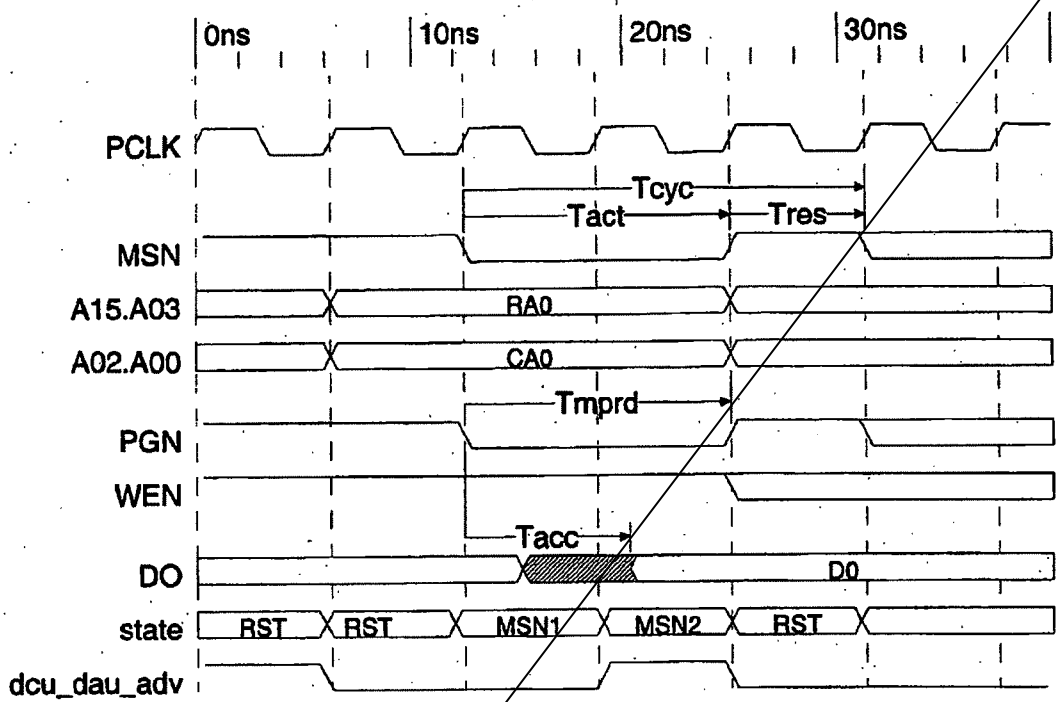


FIG. 118

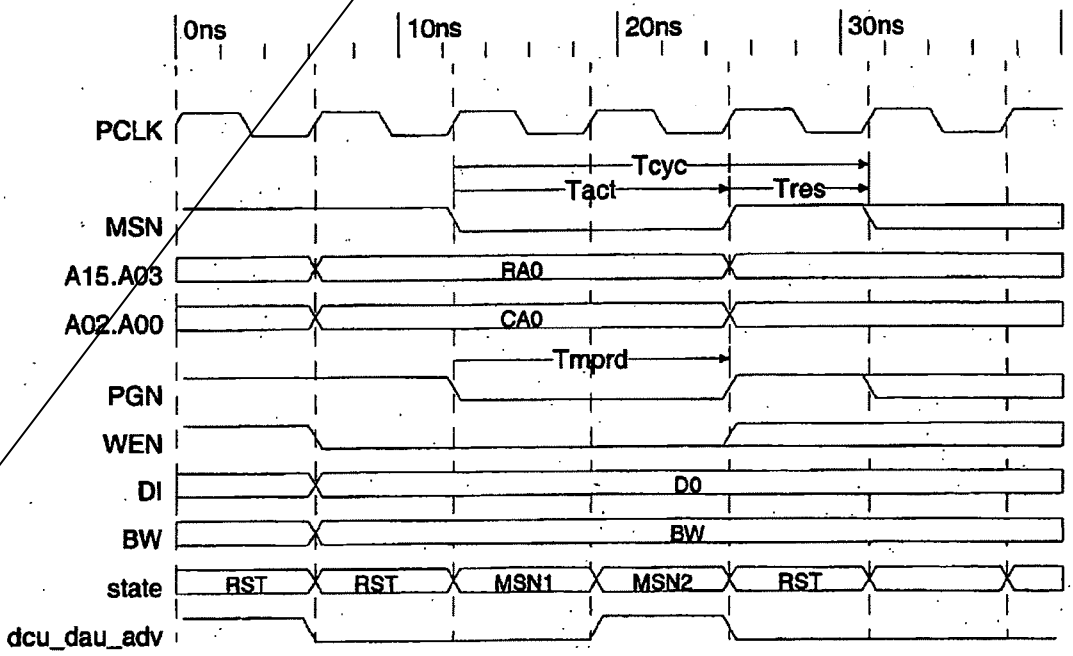


FIG. 119

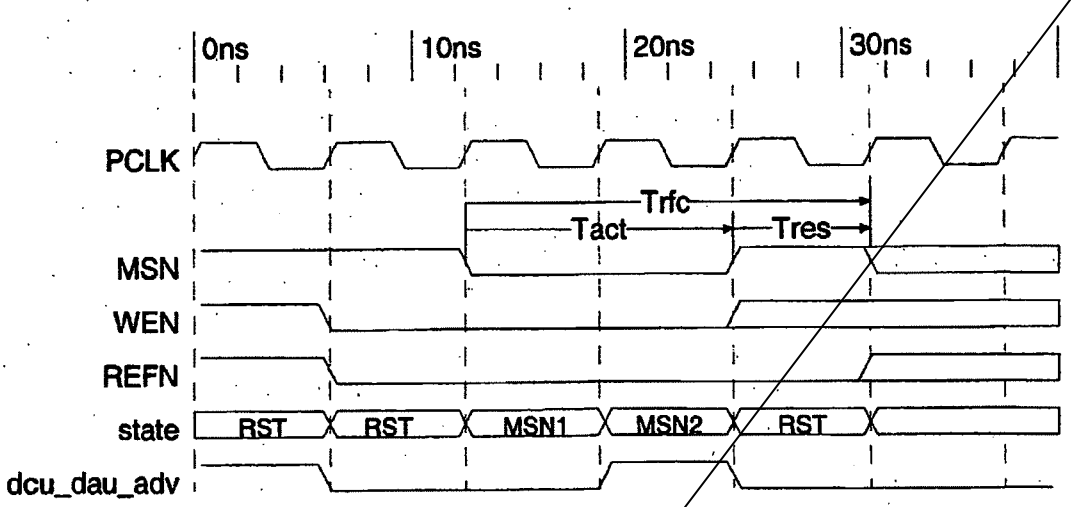


FIG. 120

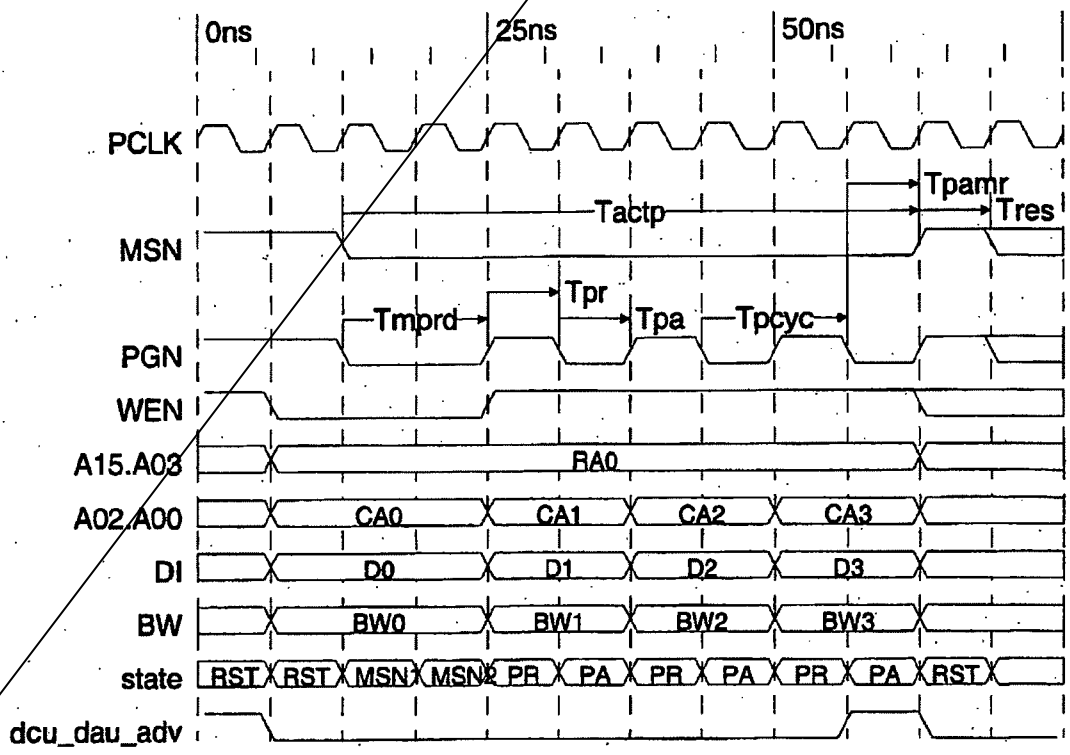


FIG. 121

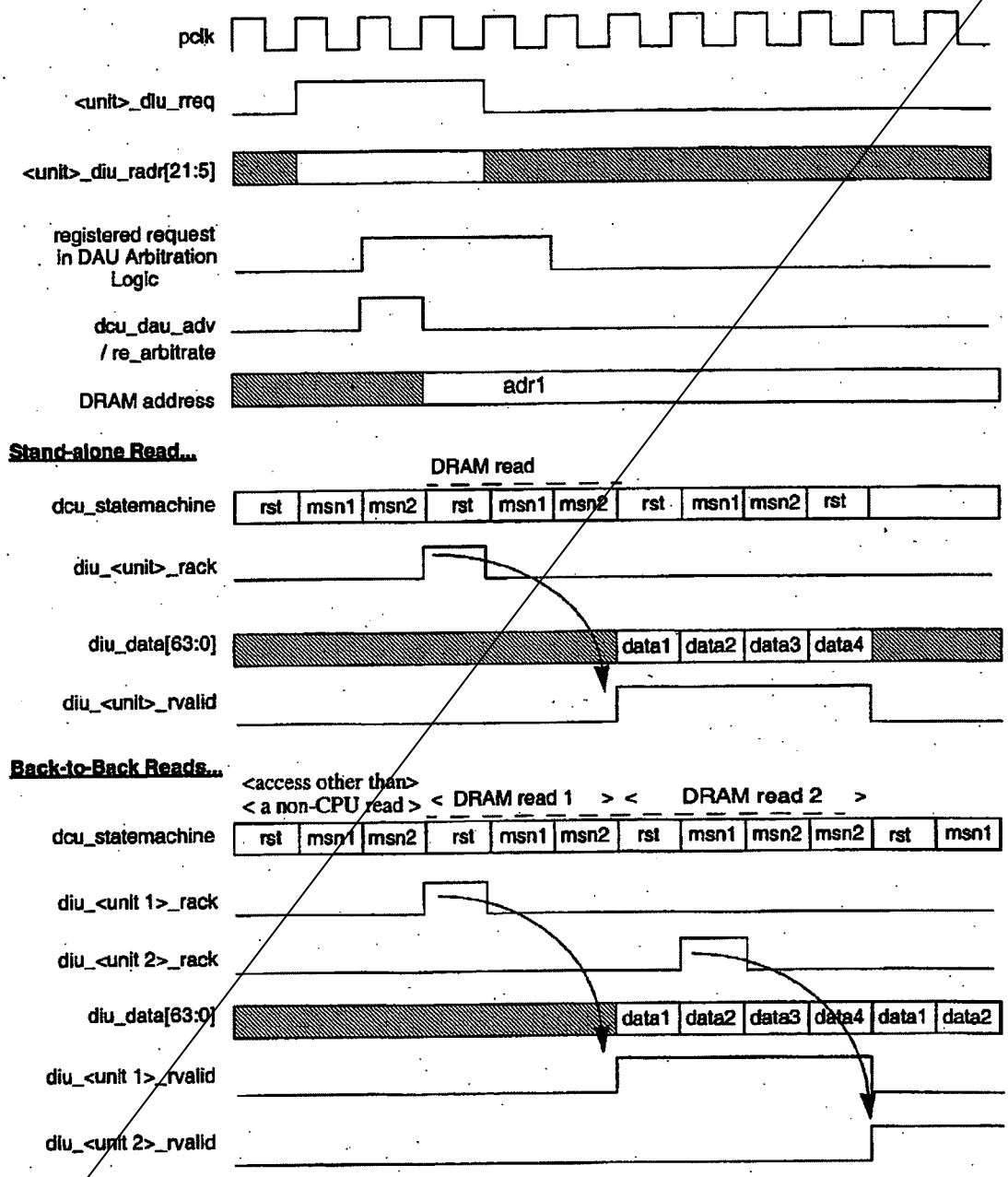

pclk
<unit>_diu_rreq
<unit>_diu_radr[21:5]
registered request in DAU Arbitration Logic
dcu_dau_adv / re_arbitrate
DRAM address
adr1
Stand-alone Read...
DRAM read
dcu_statemachine
rst
msn1
msn2
rst
msn1
msn2
rst
msn1
msn2
rst
diu_<unit>_rack
diu_data[63:0]
data1
data2
data3
data4
diu_<unit>_rvalid
Back-to-Back Reads...
<access other than>
< a non-CPU read >
< DRAM read 1 >
< DRAM read 2 >
dcu_statemachine
rst
msn1
msn2
rst
msn1
msn2
rst
msn1
msn2
msn2
rst
msn1
diu_<unit 1>_rack
diu_<unit 2>_rack
diu_data[63:0]
data1
data2
data3
data4
data1
data2
diu_<unit 1>_rvalid
diu_<unit 2>_rvalid

FIG. 122

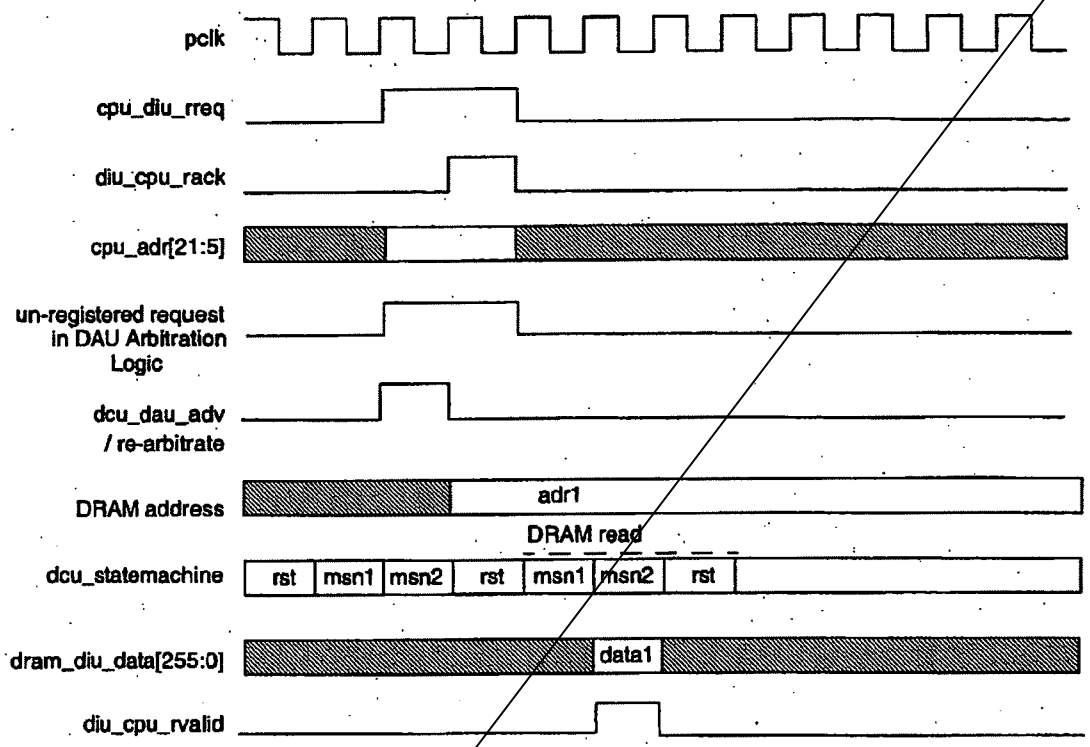


FIG. 123

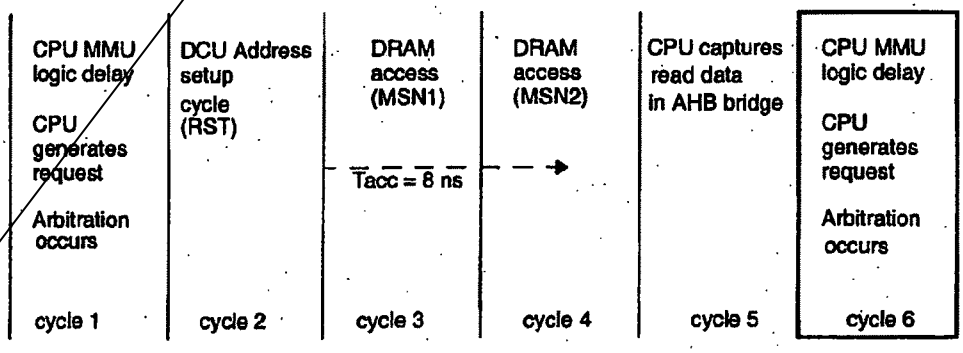


FIG. 124

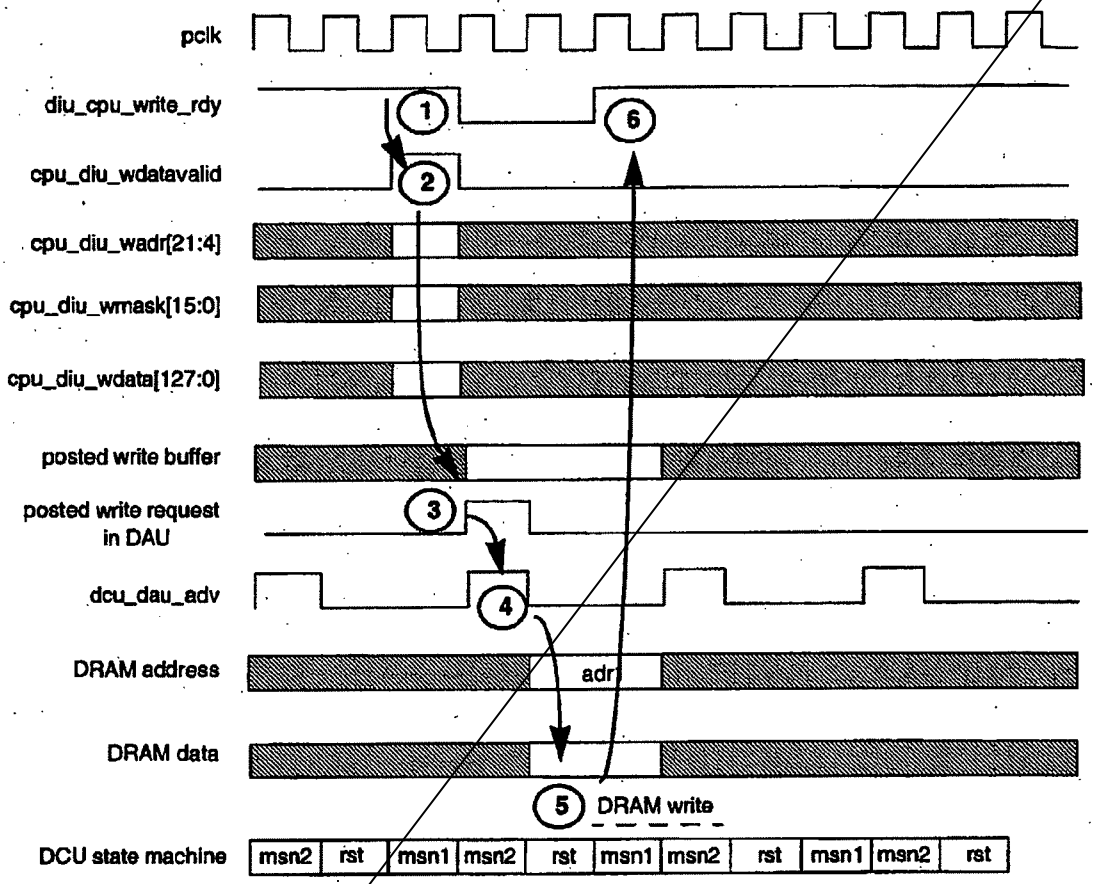

pclk
diu_cpu_write_rdy
cpu_diu_wdatavalid
cpu_diu_wadr[21:4]
cpu_diu_wmask[15:0]
cpu_diu_wdata[127:0]
posted write buffer
posted write request in DAU
dcu_dau_adv
DRAM address
DRAM data
DCU state machine
1
2
3
4
5
6
adr
DRAM write
msn2
rst
msn1
msn2
rst
msn1
msn2
rst
msn1
msn2
rst

FIG. 125

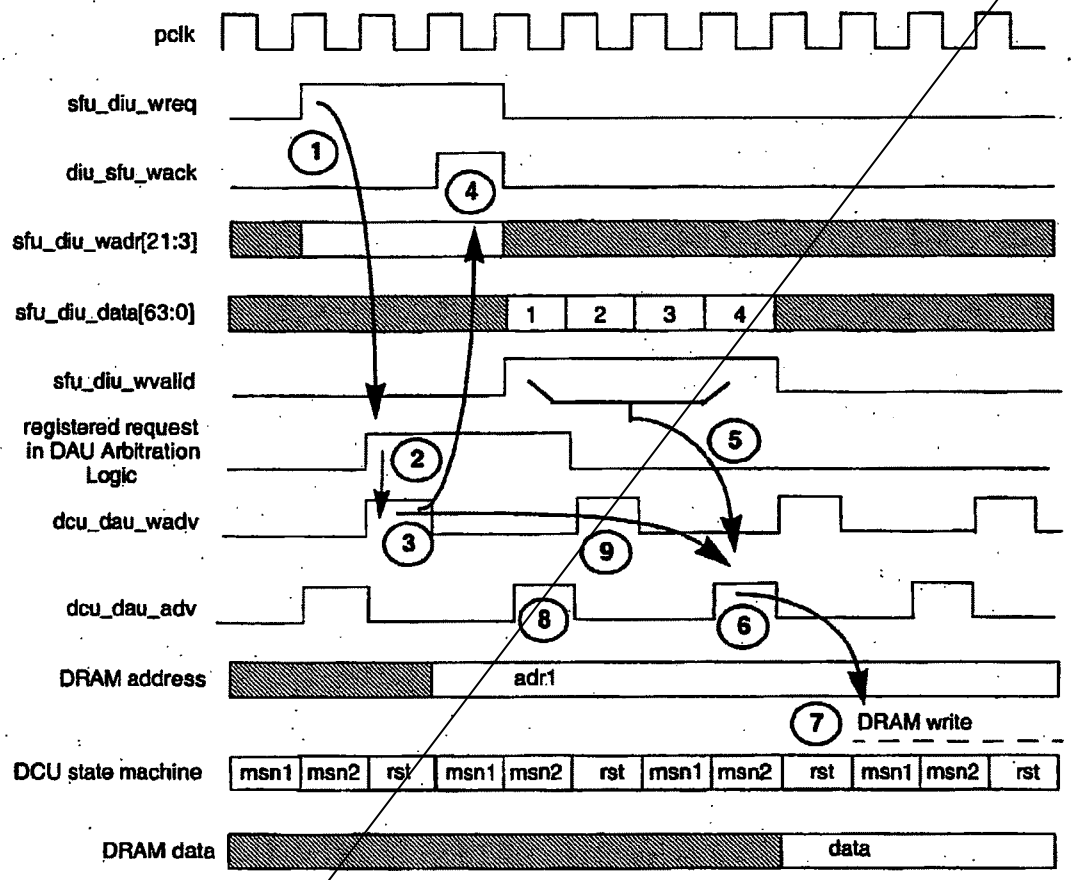

pclk
sfu_diu_wreq
diu_sfu_wack
sfu_diu_wadr[21:3]
sfu_diu_data[63:0]
1
2
3
4
sfu_diu_wvalid
registered request in DAU Arbitration Logic
dcu_dau_wadv
dcu_dau_adv
DRAM address
adr1
DRAM write
DCU state machine
msn1
msn2
rst
DRAM data
data

FIG. 126

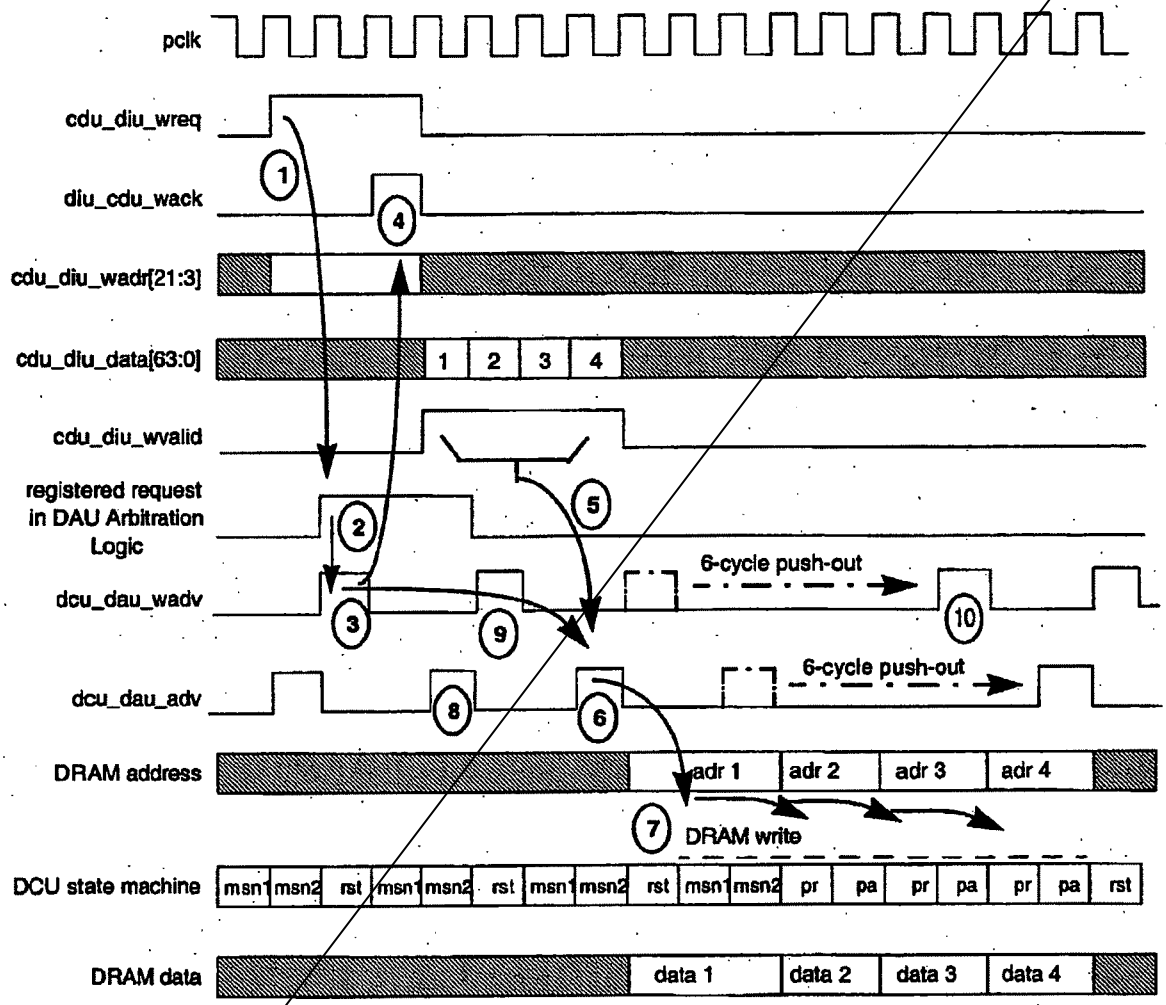

pclk
cdu_diu_wreq
diu_cdu_wack
cdu_diu_wadr[21:3]
cdu_diu_data[63:0]
1
2
3
4
cdu_diu_wvalid
registered request in DAU Arbitration Logic
dcu_dau_wadv
6-cycle push-out
dcu_dau_adv
6-cycle push-out
DRAM address
adr 1
adr 2
adr 3
adr 4
DRAM write
DCU state machine
msn1 msn2 rst msn1 msn2 rst msn1 msn2 rst msn1 msn2 pr pa pr pa pr pa rst
DRAM data
data 1
data 2
data 3
data 4

FIG. 127

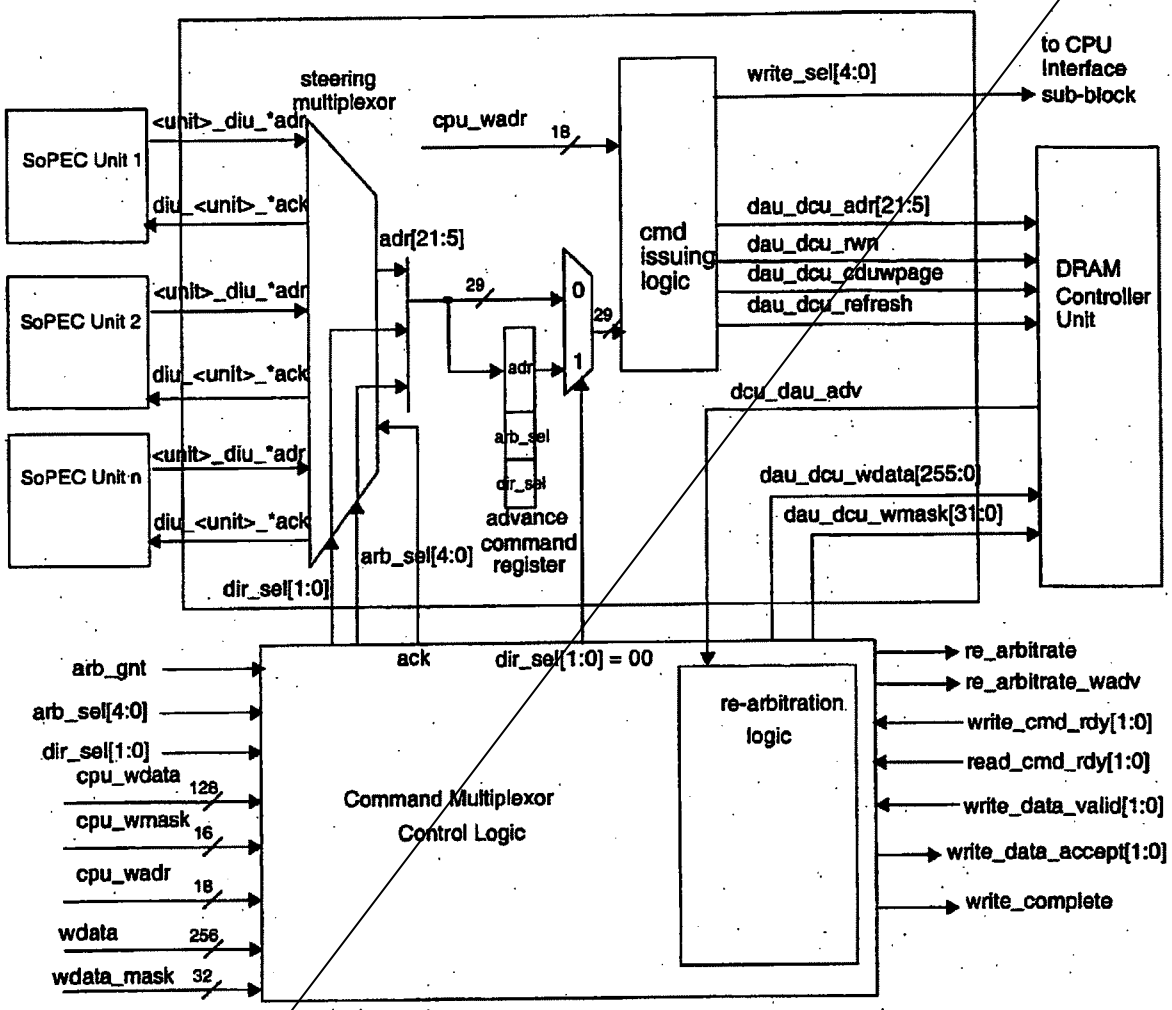

SoPEC Unit 1
SoPEC Unit 2
SoPEC Unit n
<unit>_diu_*adr
diu_<unit>_*ack
steering multiplexor
cpu_wadr
18
adr[21:5]
29
0
1
adr
arb_sel
dir_sel
advance command register
arb_sel[4:0]
dir_sel[1:0]
cmd issuing logic
write_sel[4:0]
to CPU Interface sub-block
dau_dcu_adr[21:5]
dau_dcu_rwn
dau_dcu_cduwpage
dau_dcu_refresh
dcu_dau_adv
dau_dcu_wdata[255:0]
dau_dcu_wmask[31:0]
DRAM Controller Unit
ack
dir_sel[1:0] = 00
arb_gnt
arb_sel[4:0]
dir_sel[1:0]
cpu_wdata
128
cpu_wmask
16
cpu_wadr
18
wdata
256
wdata_mask
32
Command Multiplexor Control Logic
re-arbitration logic
re_arbitrate
re_arbitrate_wadv
write_cmd_rdy[1:0]
read_cmd_rdy[1:0]
write_data_valid[1:0]
write_data_accept[1:0]
write_complete

FIG. 128

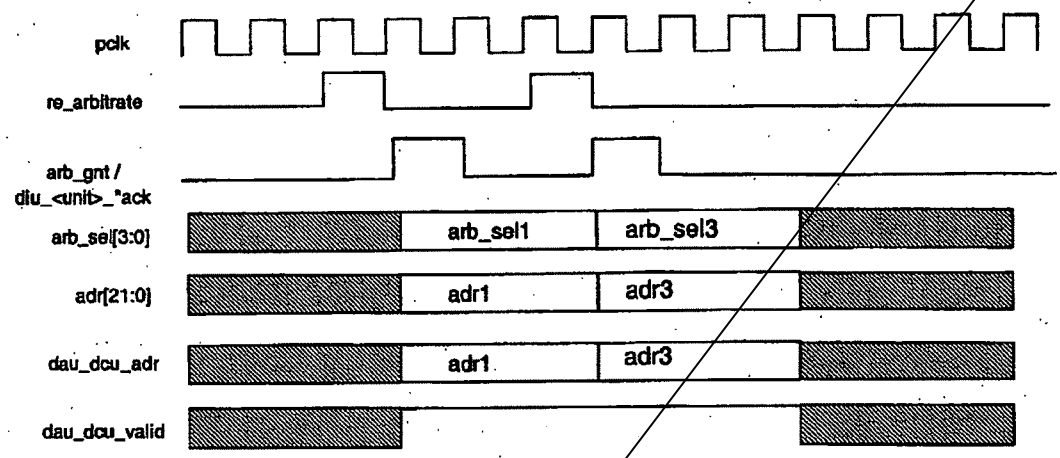


FIG. 129

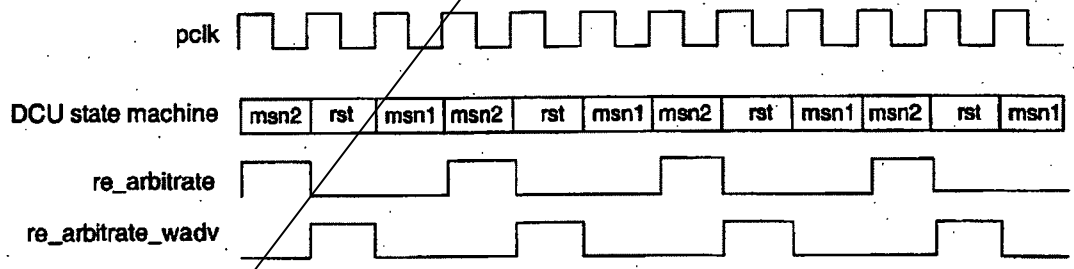


FIG. 130

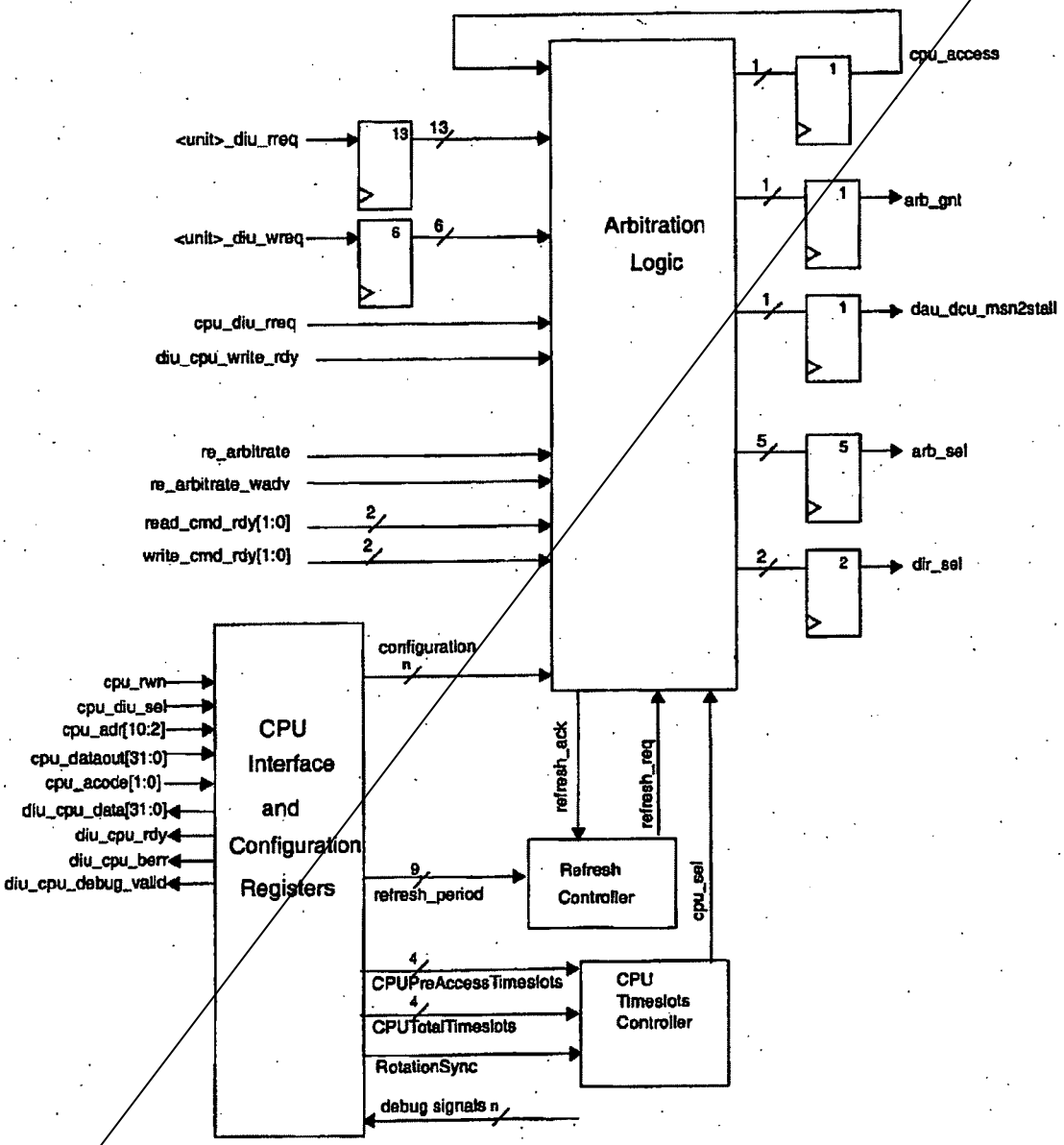

cpu_access
<unit>_diu_rreq
13
<unit>_diu_wreq
6
Arbitration Logic
arb_gnt
dau_dcu_msn2stall
cpu_diu_rreq
diu_cpu_write_rdy
re_arbitrate
re_arbitrate_wadv
read_cmd_rdy[1:0]
write_cmd_rdy[1:0]
arb_sel
dir_sel
configuration
n
cpu_rwn
cpu_diu_sel
cpu_adr[10:2]
cpu_dataout[31:0]
cpu_acode[1:0]
diu_cpu_data[31:0]
diu_cpu_rdy
diu_cpu_berr
diu_cpu_debug_valid
CPU Interface and Configuration Registers
refresh_ack
refresh_req
cpu_sel
refresh_period
Refresh Controller
CPUPreAccessTimeslots
CPUTotalTimeslots
RotationSync
CPU Timeslots Controller
debug signals n

FIG. 131

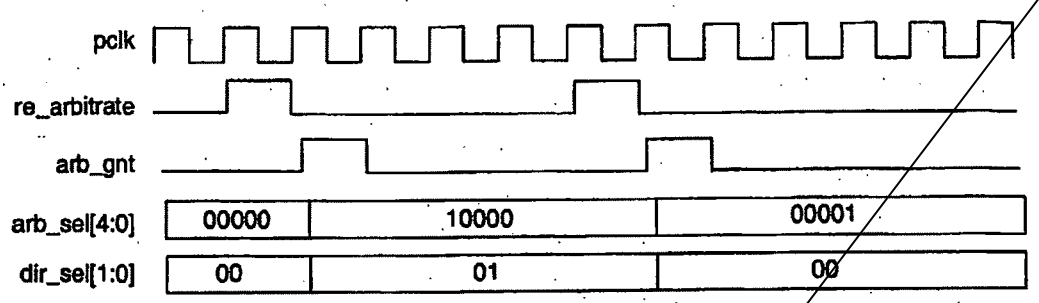


FIG. 132

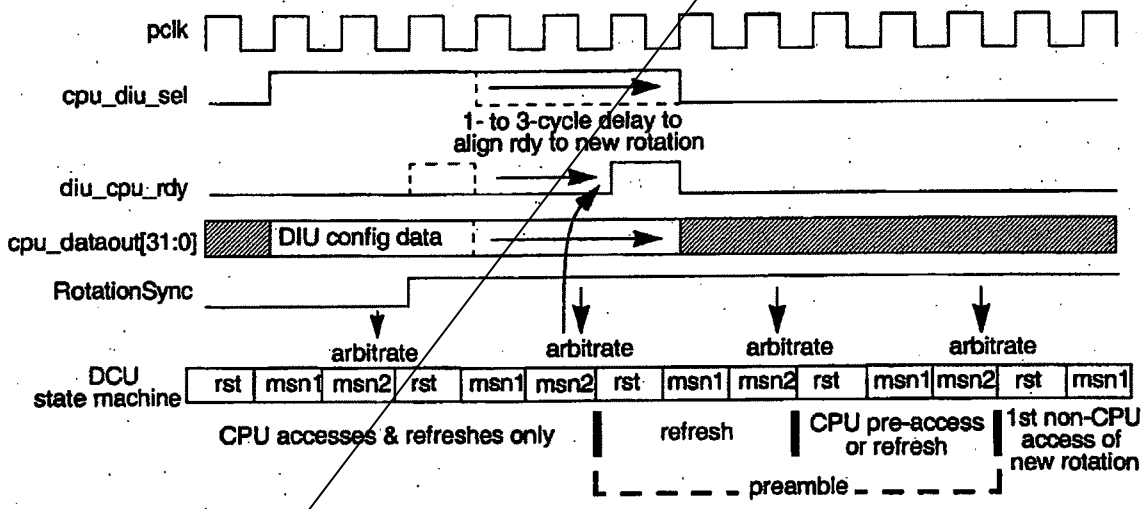


FIG. 133

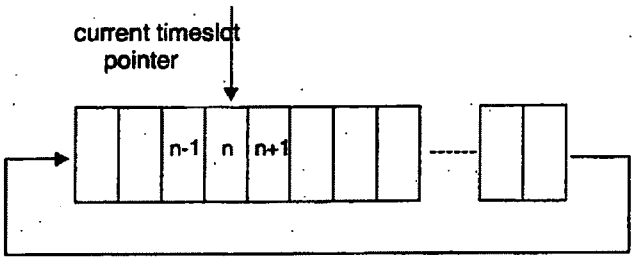


FIG. 134

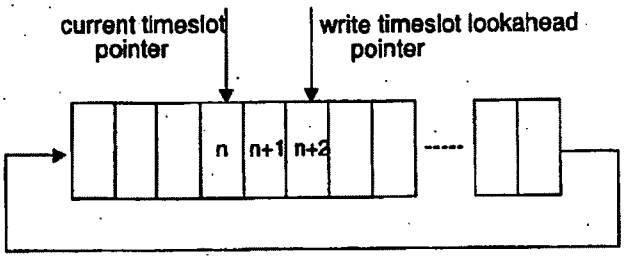


FIG. 135

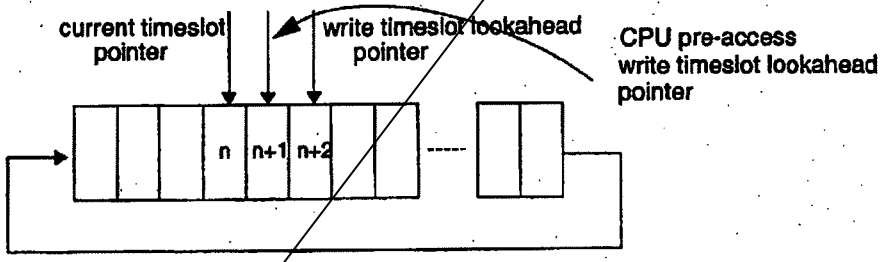


FIG. 136

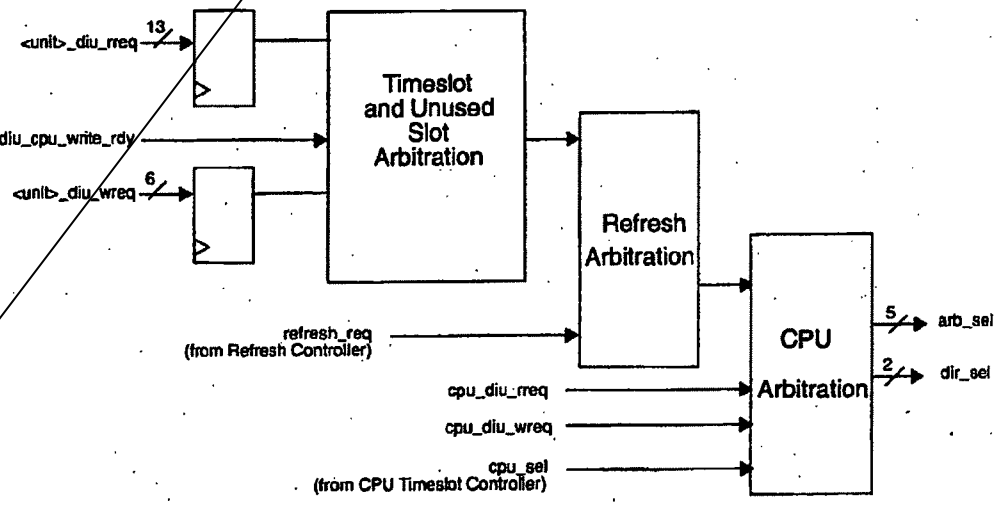


FIG. 137

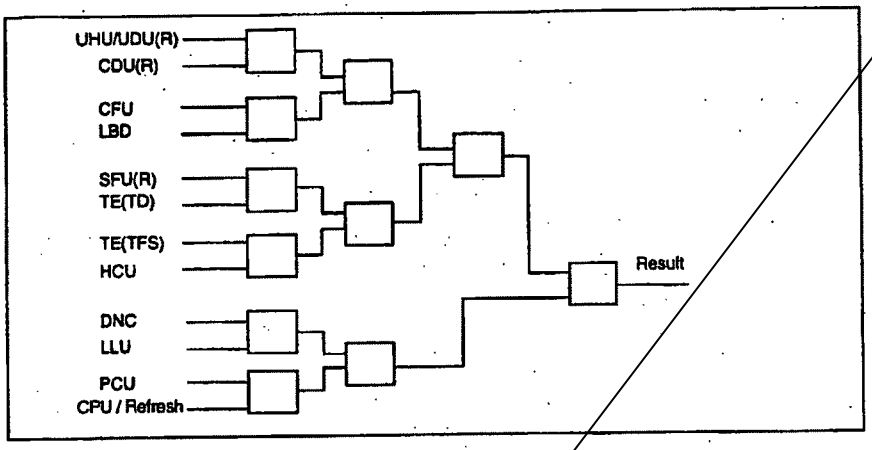


FIG. 138

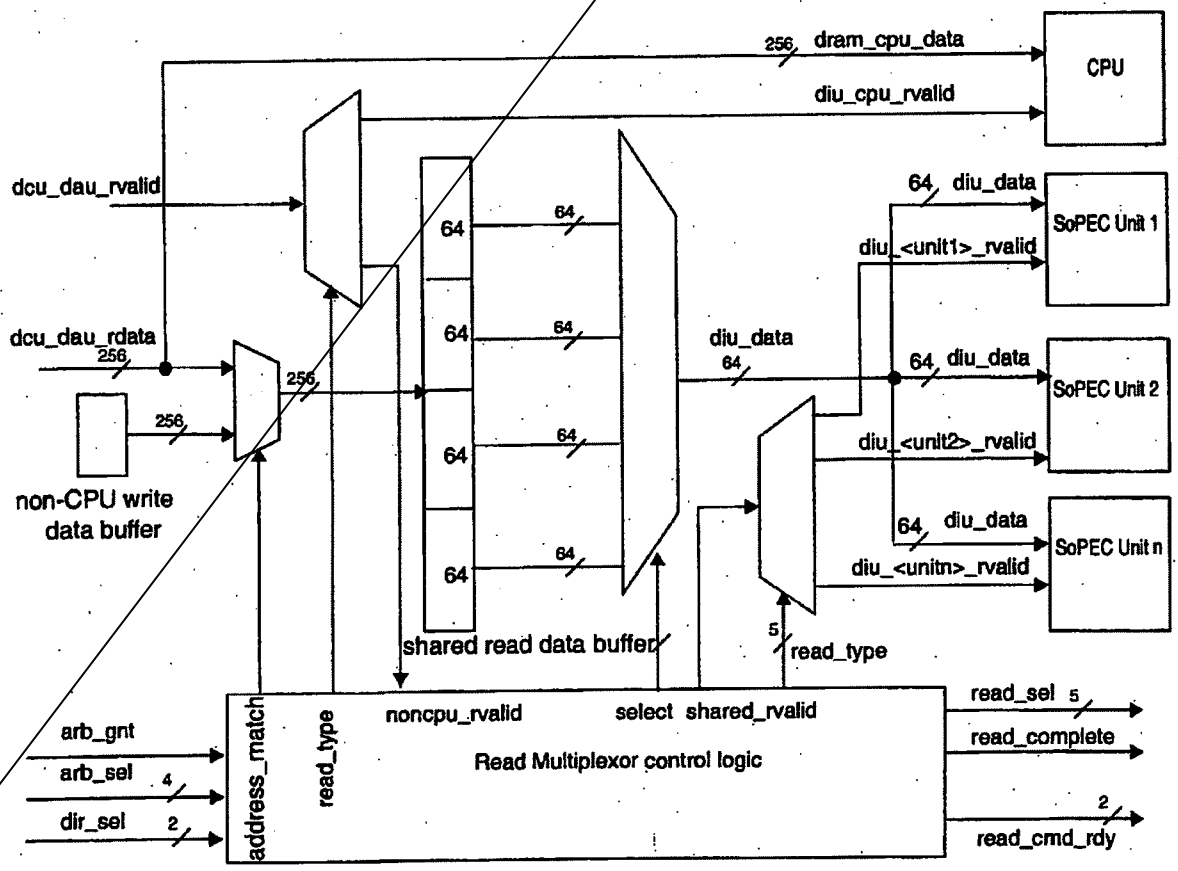


FIG. 139

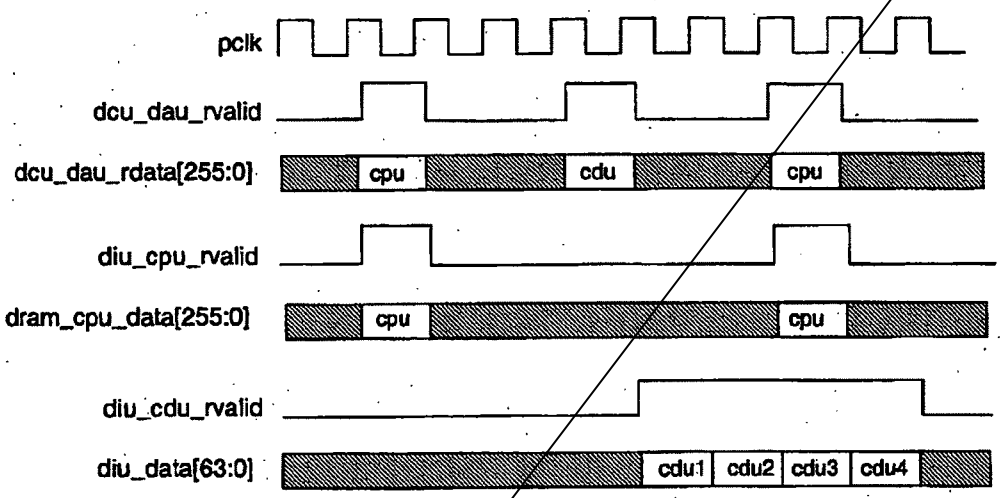


FIG. 140

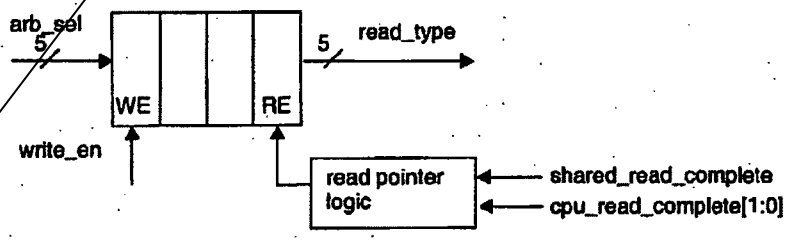


FIG. 141

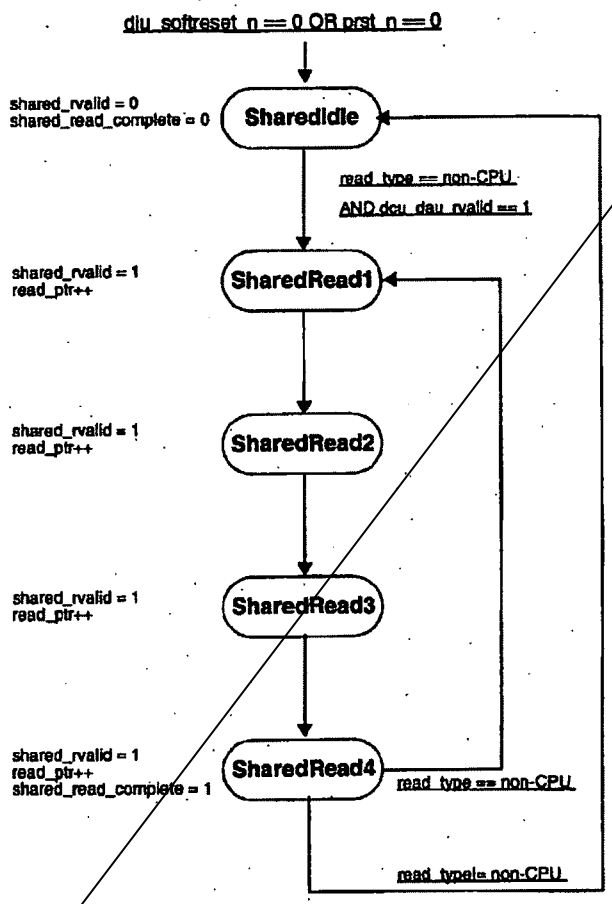

diu_softreset_n == 0 OR prst_n == 0
shared_rvalid = 0
shared_read_complete = 0
SharedIdle
read_type == non-CPU
AND dcu_dau_rvalid == 1
shared_rvalid = 1
read_ptr++
SharedRead1
shared_rvalid = 1
read_ptr++
SharedRead2
shared_rvalid = 1
read_ptr++
SharedRead3
shared_rvalid = 1
read_ptr++
shared_read_complete = 1
SharedRead4
read_type == non-CPU
read_type!= non-CPU

FIG. 142

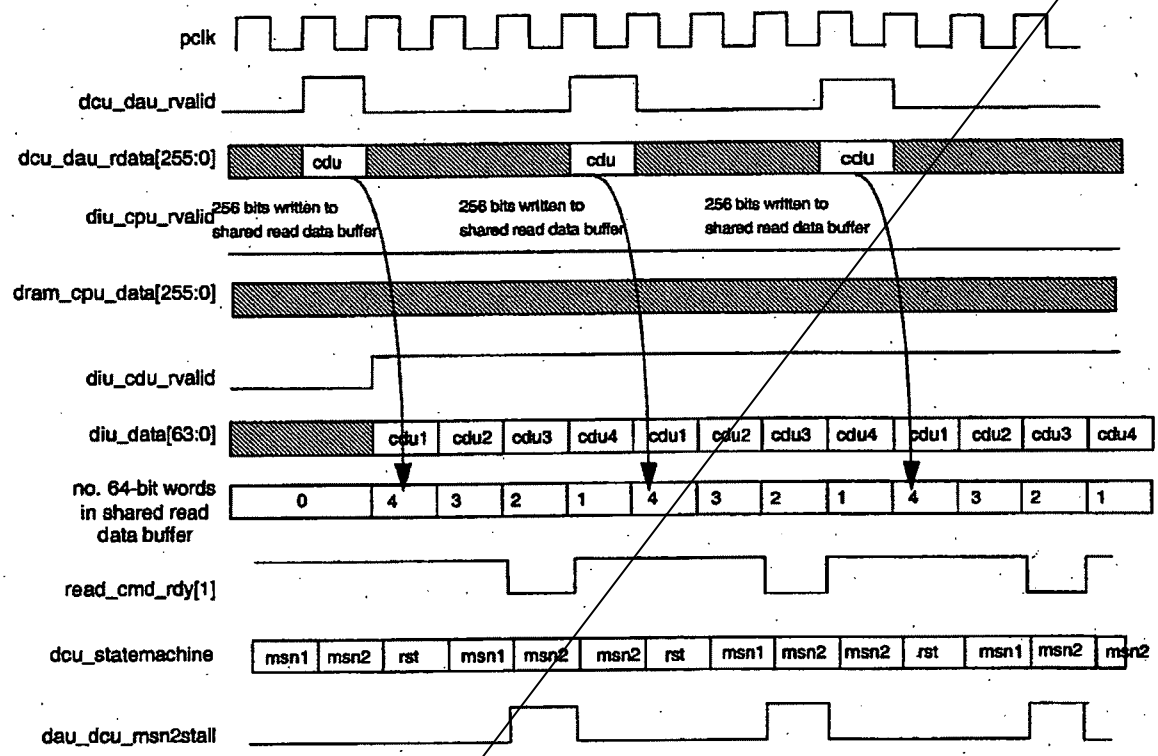

pclk
dcu_dau_rvalid
dcu_dau_rdata[255:0]
cdu
cdu
cdu
diu_cpu_rvalid
256 bits written to shared read data buffer
256 bits written to shared read data buffer
256 bits written to shared read data buffer
dram_cpu_data[255:0]
diu_cdu_rvalid
diu_data[63:0]
cdu1 cdu2 cdu3 cdu4 cdu1 cdu2 cdu3 cdu4 cdu1 cdu2 cdu3 cdu4
no. 64-bit words in shared read data buffer
0 4 3 2 1 4 3 2 1 4 3 2 1
read_cmd_rdy[1]
dcu_statemachine
msn1 msn2 rst msn1 msn2 msn2 rst msn1 msn2 msn2 rst msn1 msn2 msn2
dau_dcu_msn2stall

FIG. 143

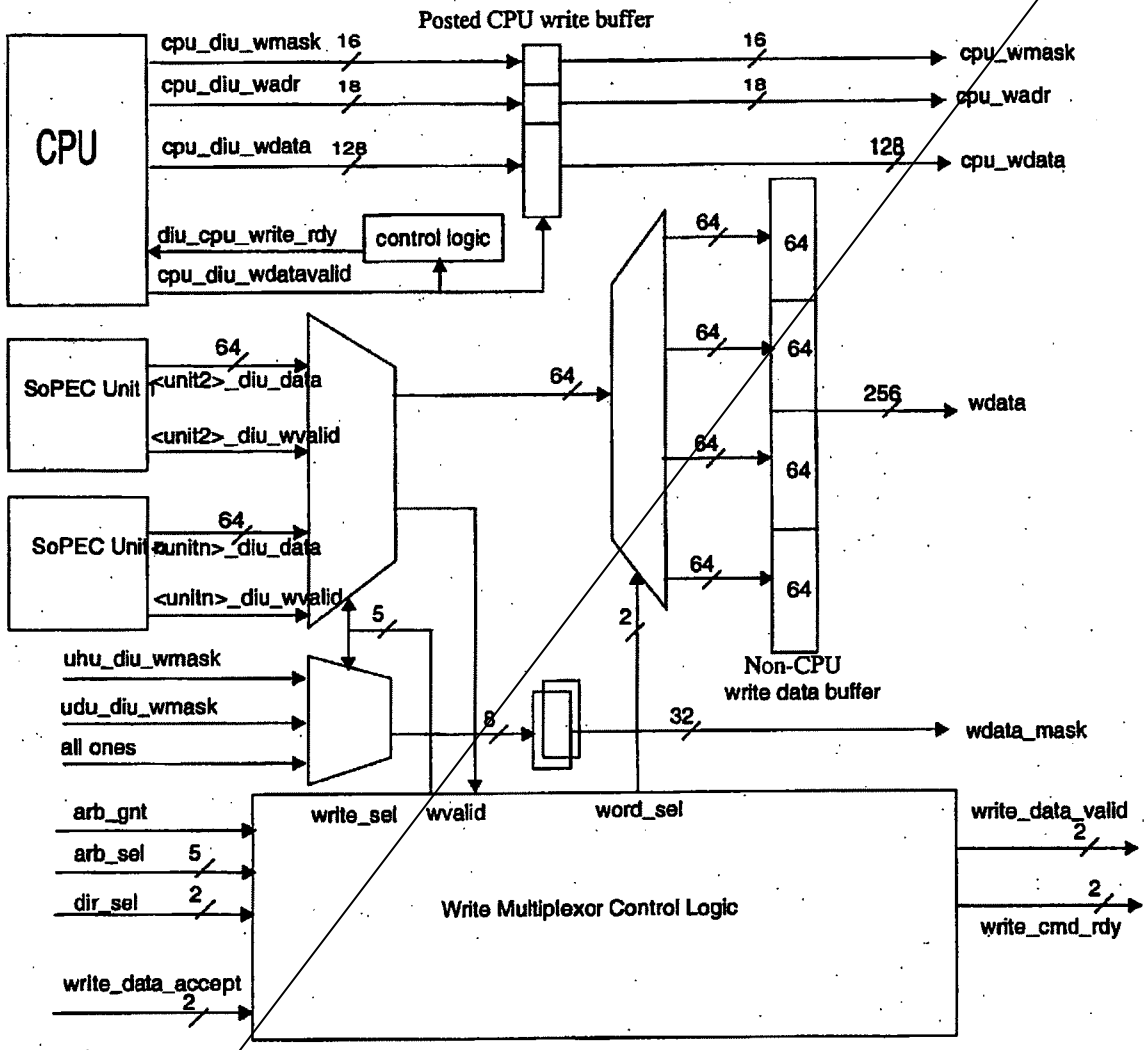

Posted CPU write buffer
CPU
cpu_diu_wmask
16
cpu_diu_wadr
18
cpu_diu_wdata
128
diu_cpu_write_rdy
cpu_diu_wdatavalid
control logic
cpu_wmask
cpu_wadr
cpu_wdata
SoPEC Unit
<unit2>_diu_data
<unit2>_diu_wvalid
SoPEC Unit
<unitn>_diu_data
<unitn>_diu_wvalid
64
256
wdata
Non-CPU
write data buffer
uhu_diu_wmask
udu_diu_wmask
all ones
5
8
32
wdata_mask
2
arb_gnt
arb_sel
dir_sel
write_data_accept
write_sel
wvalid
word_sel
Write Multiplexor Control Logic
write_data_valid
write_cmd_rdy

FIG. 144

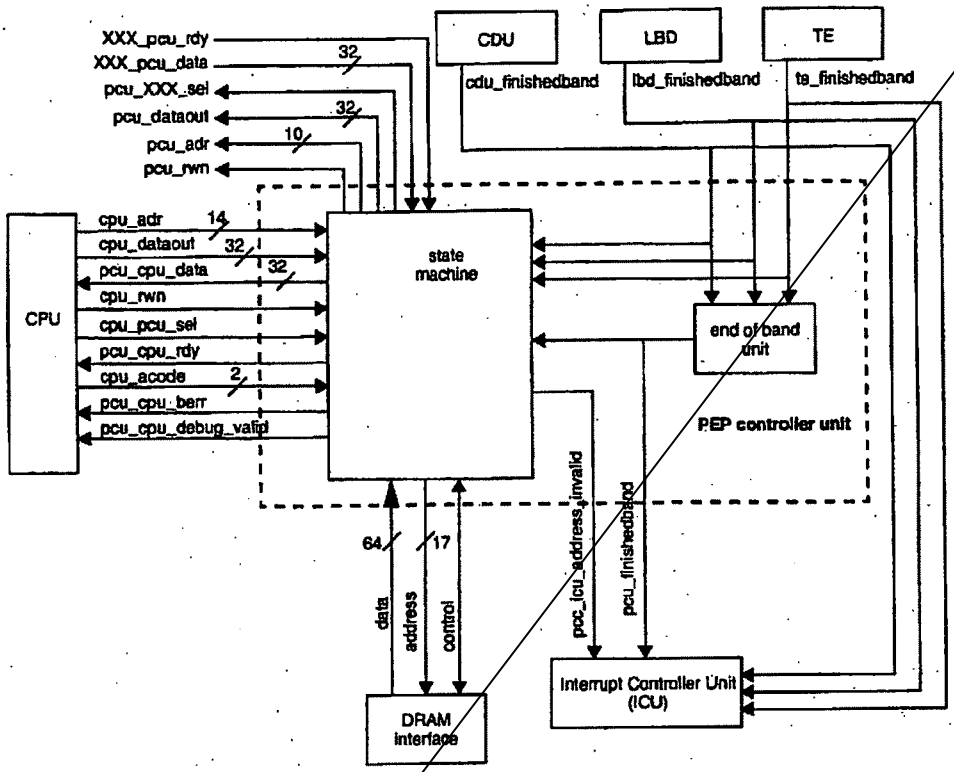


FIG. 145

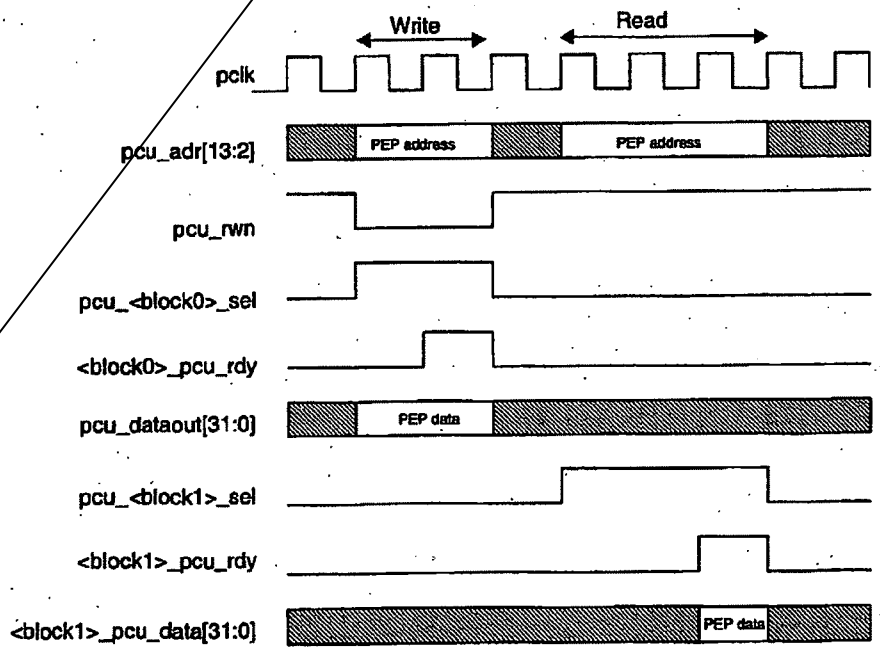


FIG. 146

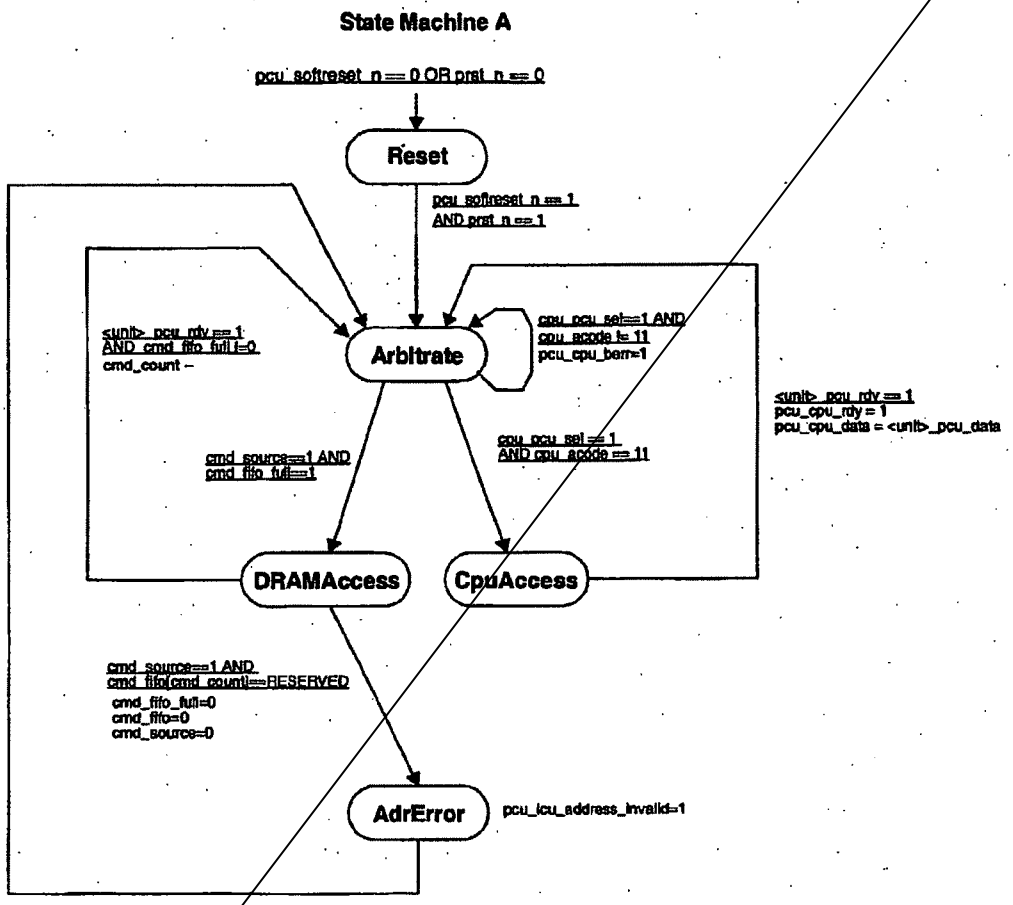

State Machine A
pcu_softreset_n == 0 OR prst_n == 0
Reset
pcu_softreset_n == 1
AND prst_n == 1
Arbitrate
<unit>_pcu_rdy == 1
AND cmd_fifo_full != 0
cmd_count --
cpu_pcu_sel==1 AND
cpu_acode != 11
pcu_cpu_ben=1
<unit>_pcu_rdy == 1
pcu_cpu_rdy = 1
pcu_cpu_data = <unit>_pcu_data
cmd_source==1 AND
cmd_fifo_full==1
cpu_pcu_sel == 1
AND cpu_acode == 11
DRAMAccess
CpuAccess
cmd_source==1 AND
cmd_fifo[cmd_count]==RESERVED
cmd_fifo_full=0
cmd_fifo=0
cmd_source=0
AdrError
pcu_icu_address_invalid=1

FIG. 147

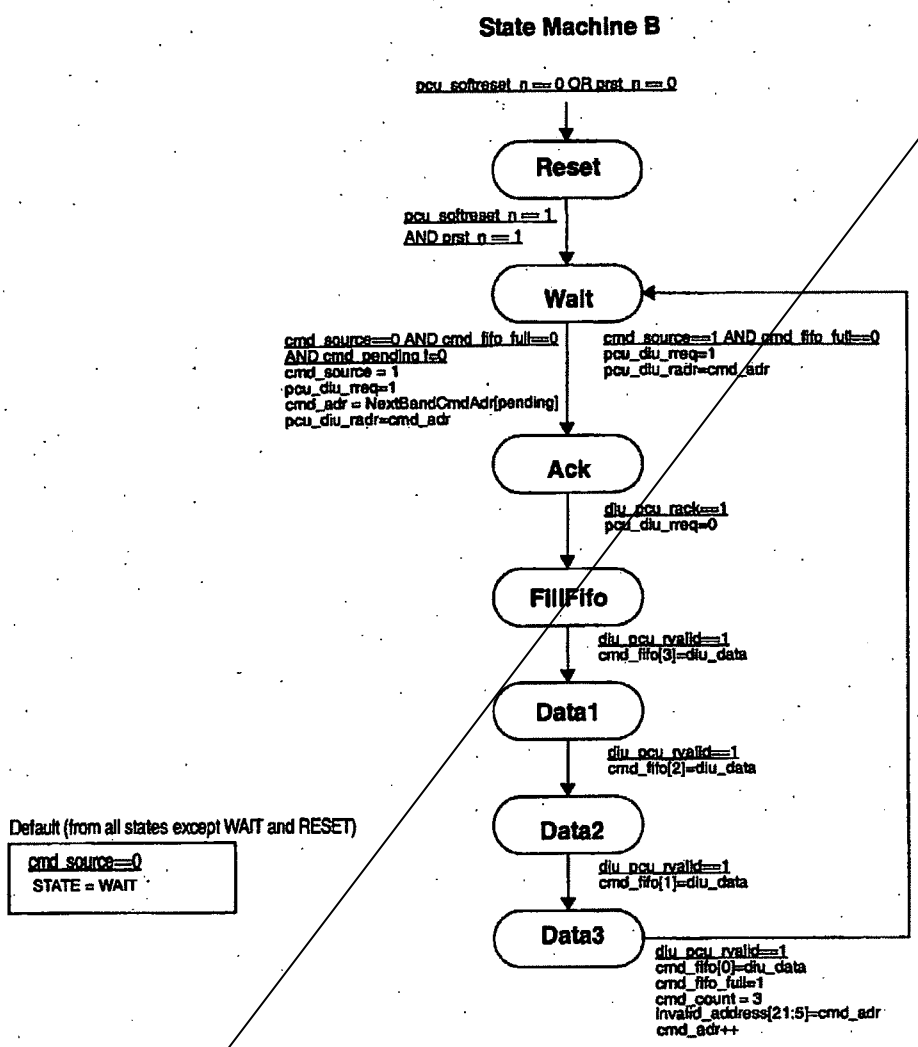


FIG. 148

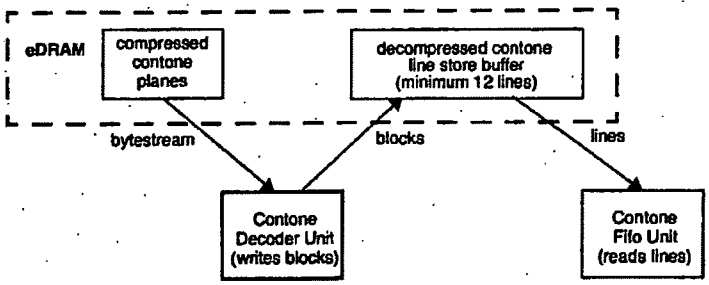


FIG. 149

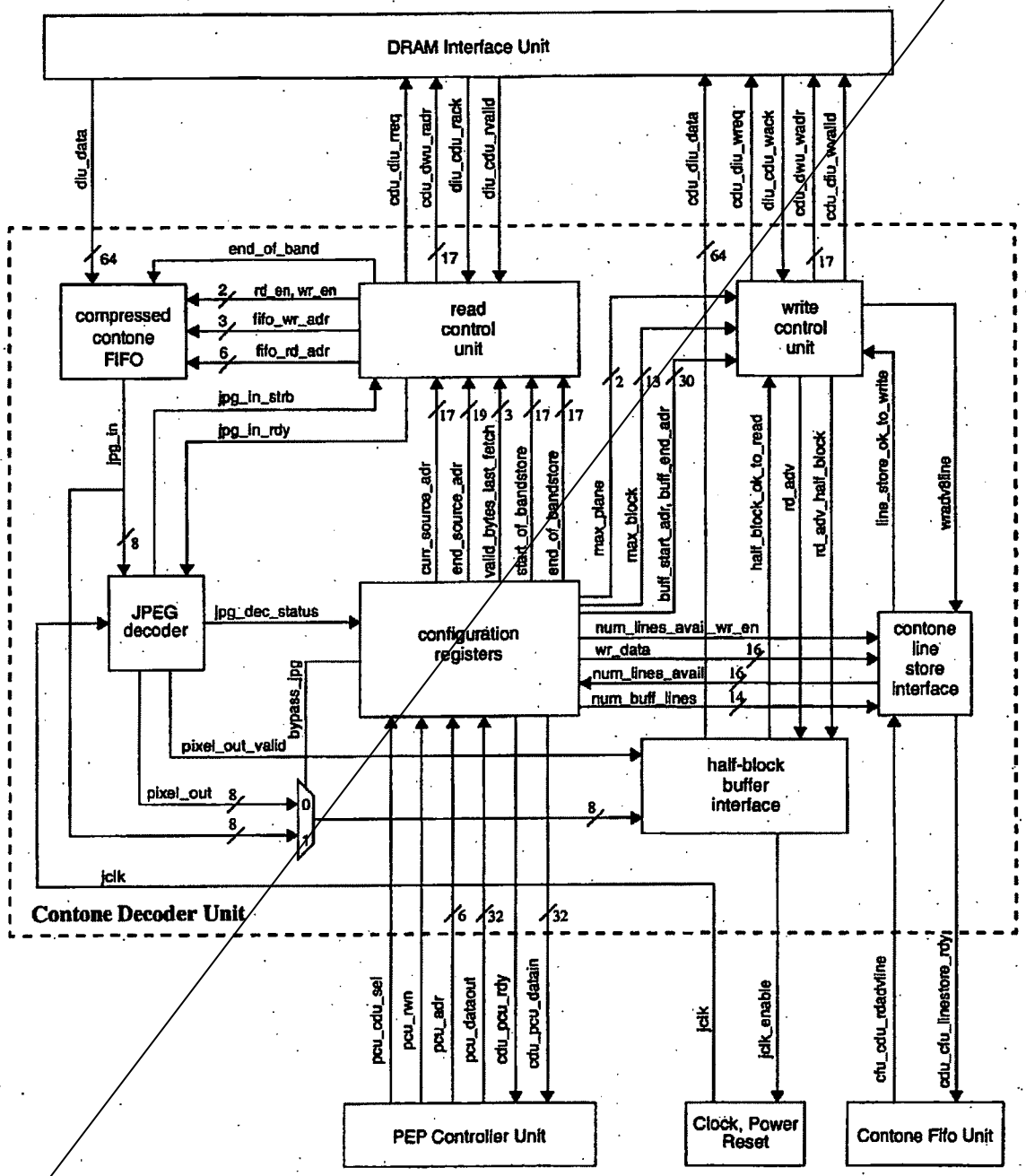

DRAM Interface Unit
diu_data
cdu_diu_req
cdu_dwu_radr
diu_cdu_rack
diu_cdu_rvalid
cdu_diu_data
cdu_diu_wreq
diu_cdu_wack
cdu_dwu_wadr
cdu_diu_wvalid
64
17
end_of_band
compressed contone FIFO
2 rd_en, wr_en
3 fifo_wr_adr
6 fifo_rd_adr
read control unit
write control unit
jpg_in_strb
jpg_in_rdy
jpg_in
8
JPEG decoder
jpg_dec_status
configuration registers
cur_source_adr
end_source_adr
valid_bytes_last_fetch
start_of_bandstore
end_of_bandstore
max_plane
max_block
buff_start_adr, buff_end_adr
half_block_ok_to_read
rd_adv
rd_adv_half_block
line_store_ok_to_write
wradv8line
num_lines_avai_wr_en
wr_data
num_lines_avail
num_buff_lines
contone line store interface
bypass_jpg
pixel_out_valid
pixel_out
half-block buffer interface
jclk
Contone Decoder Unit
pcu_cdu_sel
pcu_rwn
pcu_adr
pcu_dataout
cdu_pcu_rdy
cdu_pcu_datain
jclk_enable
cfu_cdu_rdactvline
cdu_cfu_linestore_rdy
PEP Controller Unit
Clock, Power Reset
Contone Fifo Unit

FIG. 150

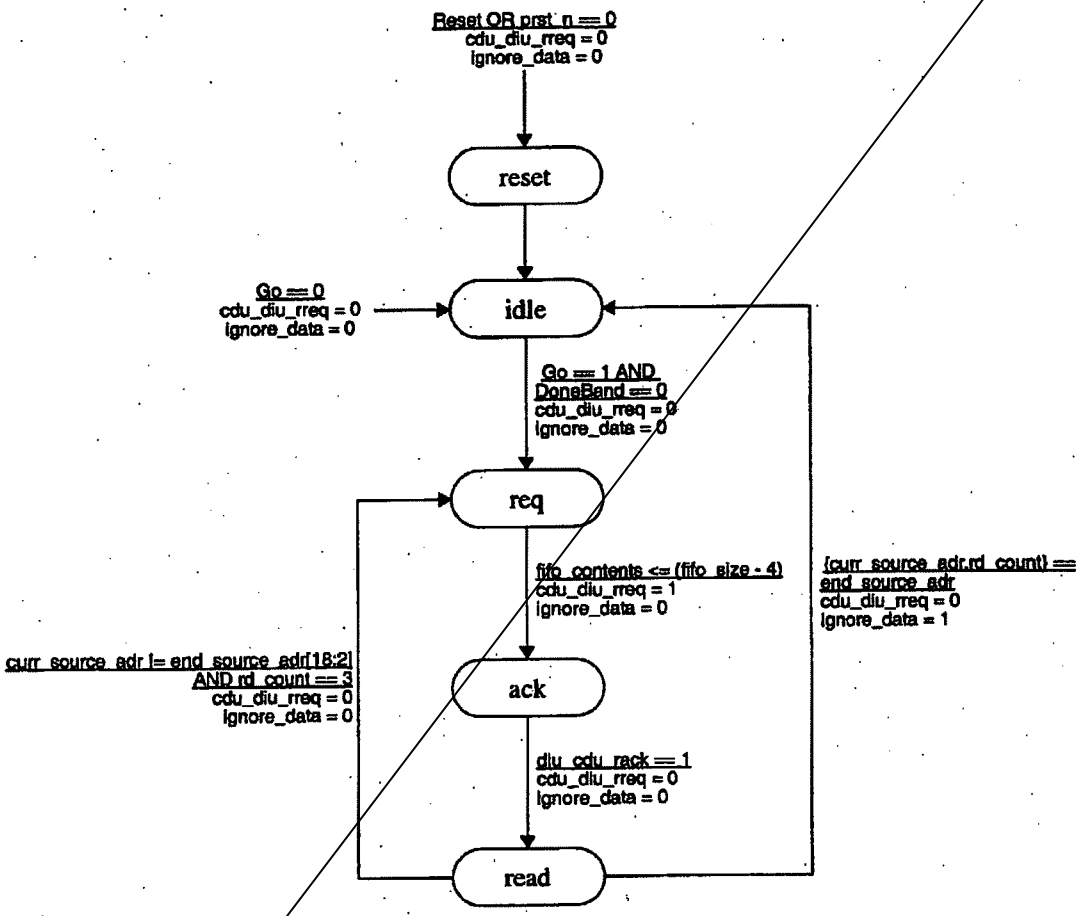

Reset OR prst_n == 0
cdu_diu_rreq = 0
ignore_data = 0
reset
Go == 0
cdu_diu_rreq = 0
ignore_data = 0
idle
Go == 1 AND
DoneBand == 0
cdu_diu_rreq = 0
ignore_data = 0
req
fifo_contents <= (fifo_size - 4)
cdu_diu_rreq = 1
ignore_data = 0
(curr_source_adr.rd_count) ==
end_source_adr
cdu_diu_rreq = 0
ignore_data = 1
curr_source_adr != end_source_adr[18:2]
AND rd_count == 3
cdu_diu_rreq = 0
ignore_data = 0
ack
diu_cdu_rack == 1
cdu_diu_rreq = 0
ignore_data = 0
read

FIG. 151

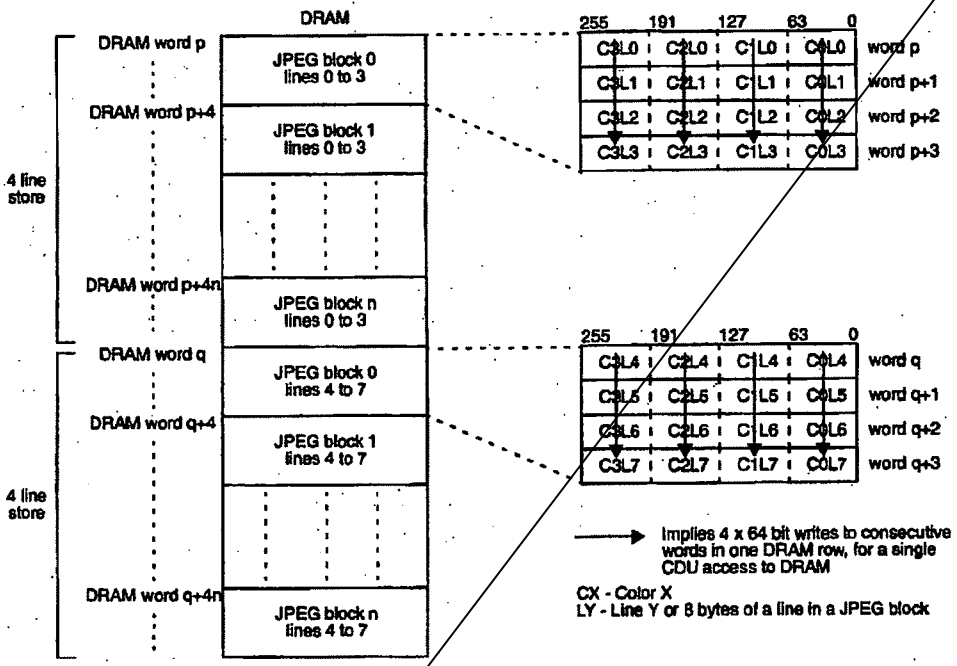

DRAM
DRAM word p
DRAM word p+4
DRAM word p+4n
4 line store
JPEG block 0 lines 0 to 3
JPEG block 1 lines 0 to 3
JPEG block n lines 0 to 3
DRAM word q
DRAM word q+4
DRAM word q+4n
4 line store
JPEG block 0 lines 4 to 7
JPEG block 1 lines 4 to 7
JPEG block n lines 4 to 7
255 191 127 63 0
C3L0 C2L0 C1L0 C0L0 word p
C3L1 C2L1 C1L1 C0L1 word p+1
C3L2 C2L2 C1L2 C0L2 word p+2
C3L3 C2L3 C1L3 C0L3 word p+3
255 191 127 63 0
C3L4 C2L4 C1L4 C0L4 word q
C3L5 C2L5 C1L5 C0L5 word q+1
C3L6 C2L6 C1L6 C0L6 word q+2
C3L7 C2L7 C1L7 C0L7 word q+3
Implies 4 x 64 bit writes to consecutive words in one DRAM row, for a single CDU access to DRAM
CX - Color X
LY - Line Y or 8 bytes of a line in a JPEG block

FIG. 152

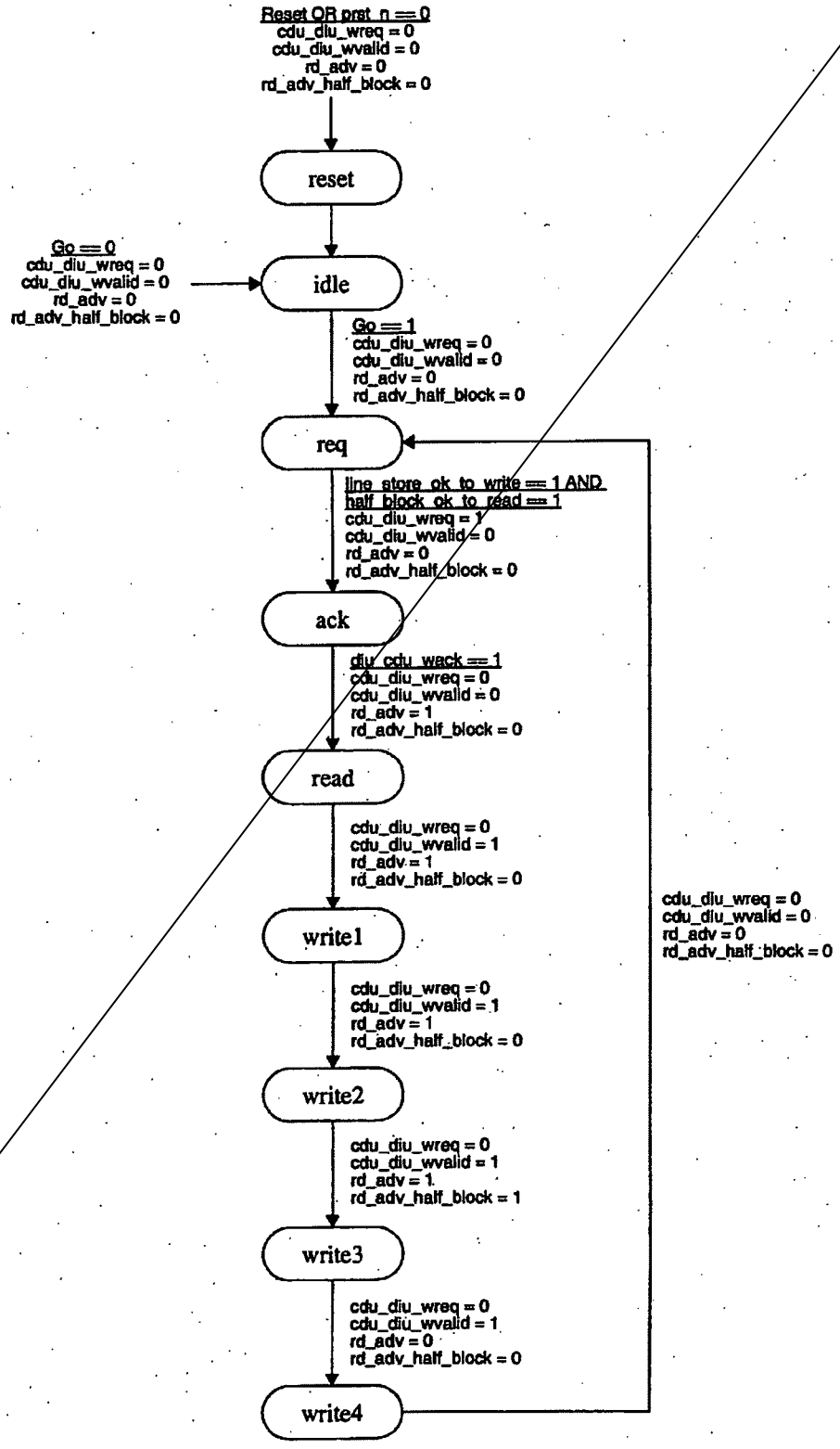

Reset OR prst_n == 0
cdu_diu_wreq = 0
cdu_diu_wvalid = 0
rd_adv = 0
rd_adv_half_block = 0
reset
Go == 0
cdu_diu_wreq = 0
cdu_diu_wvalid = 0
rd_adv = 0
rd_adv_half_block = 0
idle
Go == 1
cdu_diu_wreq = 0
cdu_diu_wvalid = 0
rd_adv = 0
rd_adv_half_block = 0
req
line_store_ok_to_write == 1 AND
half_block_ok_to_read == 1
cdu_diu_wreq = 1
cdu_diu_wvalid = 0
rd_adv = 0
rd_adv_half_block = 0
ack
diu_cdu_wack == 1
cdu_diu_wreq = 0
cdu_diu_wvalid = 0
rd_adv = 1
rd_adv_half_block = 0
read
cdu_diu_wreq = 0
cdu_diu_wvalid = 1
rd_adv = 1
rd_adv_half_block = 0
write1
cdu_diu_wreq = 0
cdu_diu_wvalid = 1
rd_adv = 1
rd_adv_half_block = 0
write2
cdu_diu_wreq = 0
cdu_diu_wvalid = 1
rd_adv = 1
rd_adv_half_block = 1
write3
cdu_diu_wreq = 0
cdu_diu_wvalid = 1
rd_adv = 0
rd_adv_half_block = 0
write4
cdu_diu_wreq = 0
cdu_diu_wvalid = 0
rd_adv = 0
rd_adv_half_block = 0

FIG. 153

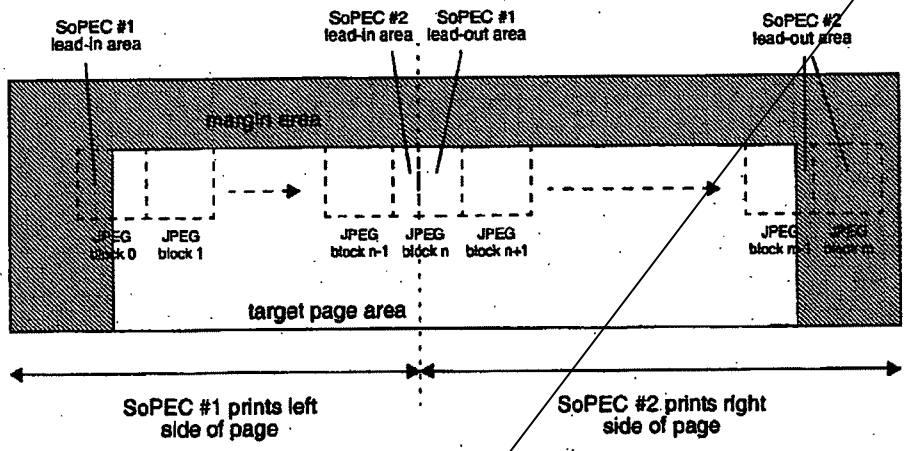

SoPEC #1
lead-in area
SoPEC #2
lead-in area
SoPEC #1
lead-out area
SoPEC #2
lead-out area
margin area
JPEG
block 0
JPEG
block 1
JPEG
block n-1
JPEG
block n
JPEG
block n+1
target page area
SoPEC #1 prints left
side of page
SoPEC #2 prints right
side of page

FIG. 154

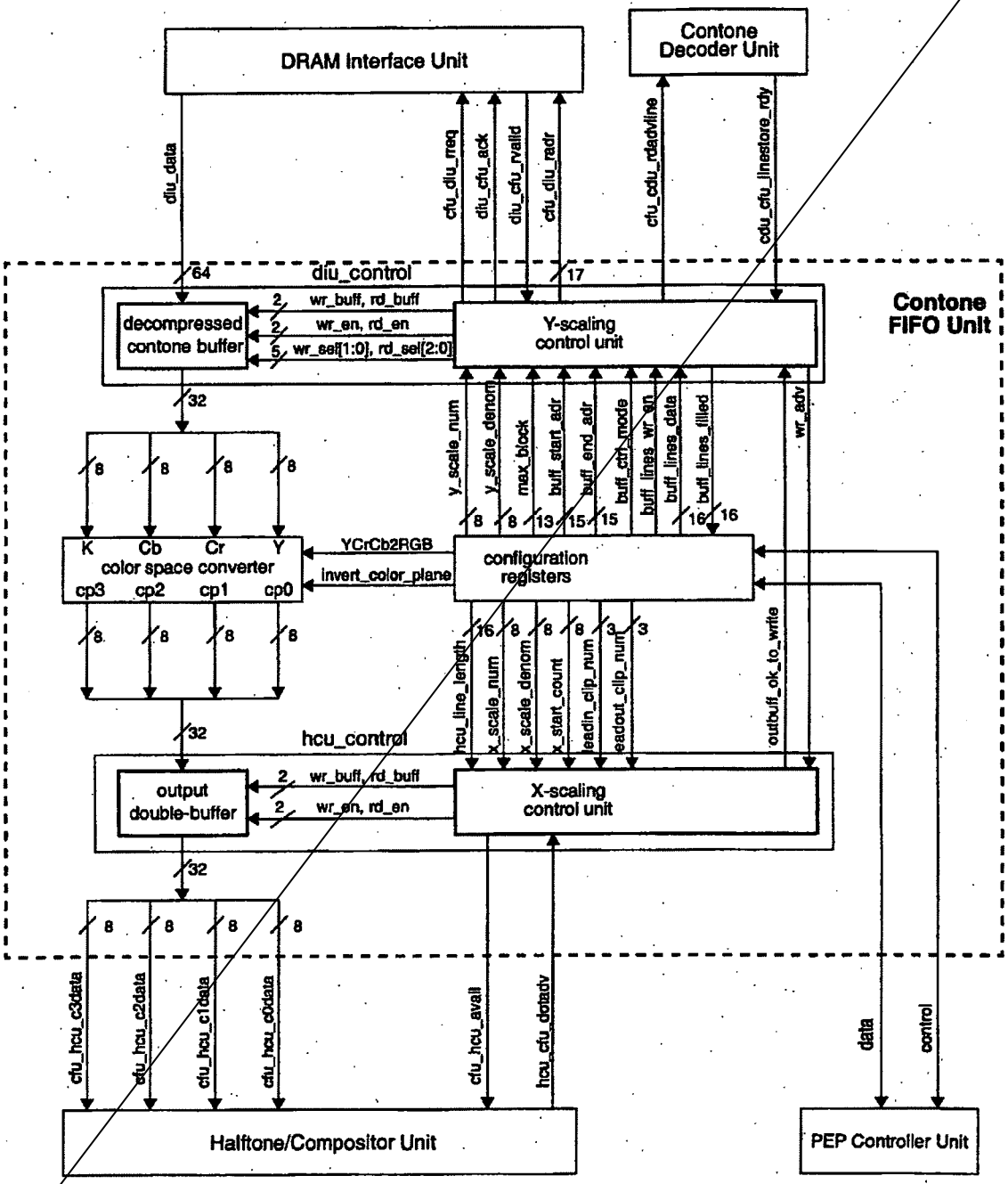

DRAM Interface Unit
Contone Decoder Unit
cfu_data
cfu_diu_req
diu_cfu_ack
diu_cfu_rvalid
cfu_diu_radr
cfu_cdu_rdadvline
cdu_cfu_linestore_rdy
Contone FIFO Unit
diu_control
64
17
decompressed contone buffer
2 wr_buff, rd_buff
2 wr_en, rd_en
5 wr_sel[1:0], rd_sel[2:0]
Y-scaling control unit
32
y_scale_num
y_scale_denom
max_block
buff_start_adr
buff_end_adr
buff_ctrl_mode
buff_lines_wr_en
buff_lines_data
buff_lines_filled
wr_adv
8
13
15
16
K Cb Cr Y
color space converter
cp3 cp2 cp1 cp0
YCrCb2RGB
invert_color_plane
configuration registers
hcu_line_length
x_scale_num
x_scale_denom
x_start_count
leadin_clip_num
leadout_clip_num
outbuff_ok_to_write
hcu_control
output double-buffer
wr_buff, rd_buff
wr_en, rd_en
X-scaling control unit
cfu_hcu_c3data
cfu_hcu_c2data
cfu_hcu_c1data
cfu_hcu_c0data
cfu_hcu_avail
hcu_cfu_dotadv
data
control
Halftone/Compositor Unit
PEP Controller Unit

FIG. 155

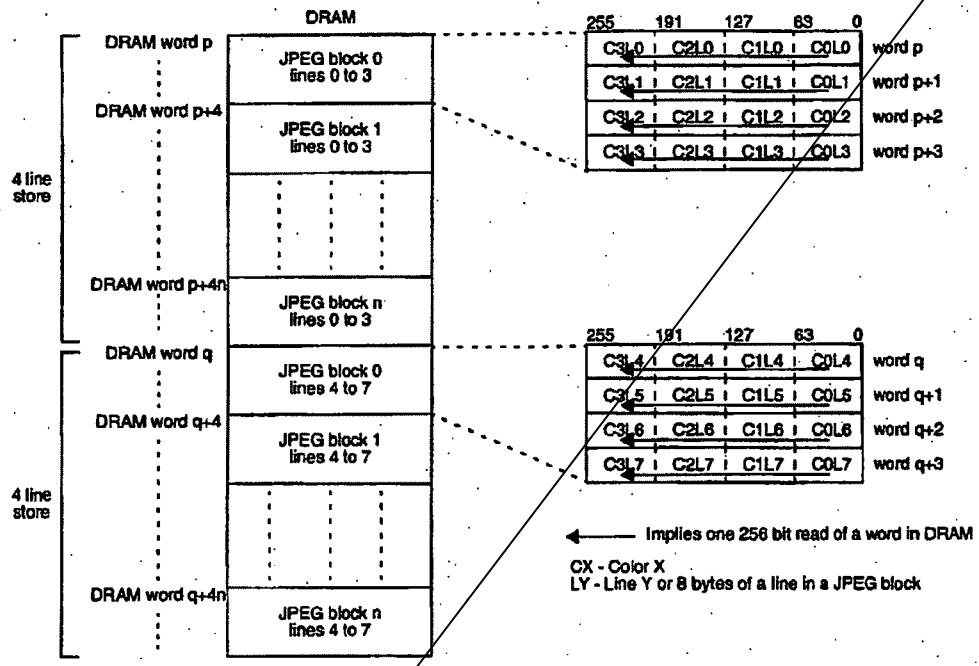

DRAM
DRAM word p
DRAM word p+4
DRAM word p+4n
4 line store
JPEG block 0 lines 0 to 3
JPEG block 1 lines 0 to 3
JPEG block n lines 0 to 3
DRAM word q
DRAM word q+4
DRAM word q+4n
4 line store
JPEG block 0 lines 4 to 7
JPEG block 1 lines 4 to 7
JPEG block n lines 4 to 7
255 191 127 63 0
C3L0 C2L0 C1L0 C0L0 word p
C3L1 C2L1 C1L1 C0L1 word p+1
C3L2 C2L2 C1L2 C0L2 word p+2
C3L3 C2L3 C1L3 C0L3 word p+3
255 191 127 63 0
C3L4 C2L4 C1L4 C0L4 word q
C3L5 C2L5 C1L5 C0L5 word q+1
C3L6 C2L6 C1L6 C0L6 word q+2
C3L7 C2L7 C1L7 C0L7 word q+3
Implies one 256 bit read of a word in DRAM
CX - Color X
LY - Line Y or 8 bytes of a line in a JPEG block

FIG. 156

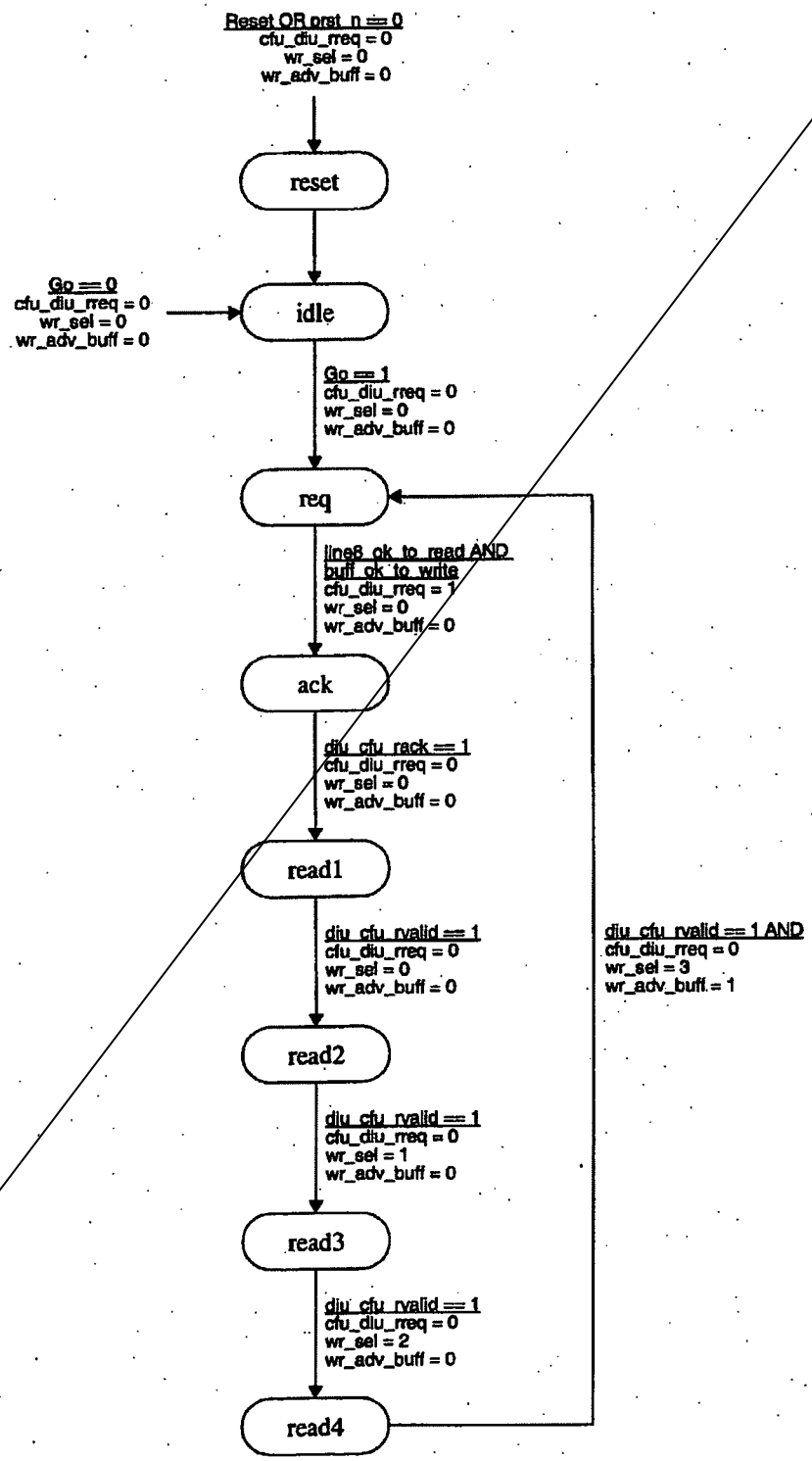

Reset OR rst_n == 0
cfu_diu_rreq = 0
wr_sel = 0
wr_adv_buff = 0
reset
Go == 0
cfu_diu_rreq = 0
wr_sel = 0
wr_adv_buff = 0
idle
Go == 1
cfu_diu_rreq = 0
wr_sel = 0
wr_adv_buff = 0
req
line8_ok_to_read AND
buff_ok_to_write
cfu_diu_rreq = 1
wr_sel = 0
wr_adv_buff = 0
ack
diu_cfu_rack == 1
cfu_diu_rreq = 0
wr_sel = 0
wr_adv_buff = 0
read1
diu_cfu_rvalid == 1
cfu_diu_rreq = 0
wr_sel = 0
wr_adv_buff = 0
read2
diu_cfu_rvalid == 1
cfu_diu_rreq = 0
wr_sel = 1
wr_adv_buff = 0
read3
diu_cfu_rvalid == 1
cfu_diu_rreq = 0
wr_sel = 2
wr_adv_buff = 0
read4
diu_cfu_rvalid == 1 AND
cfu_diu_rreq = 0
wr_sel = 3
wr_adv_buff = 1

FIG. 157

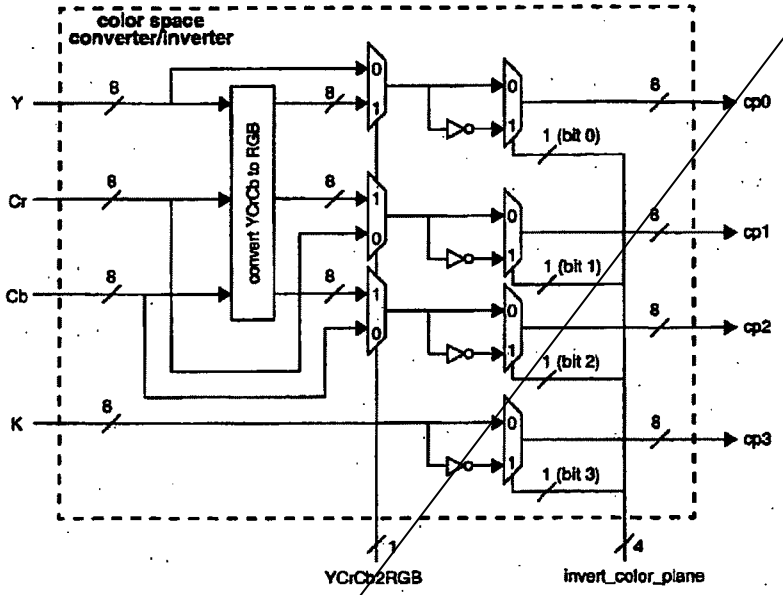


FIG. 158

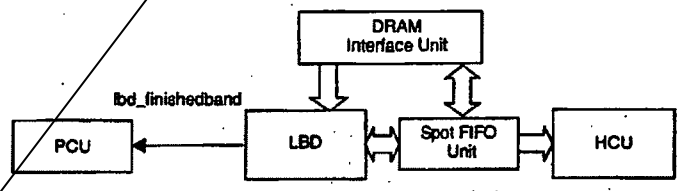


FIG. 159

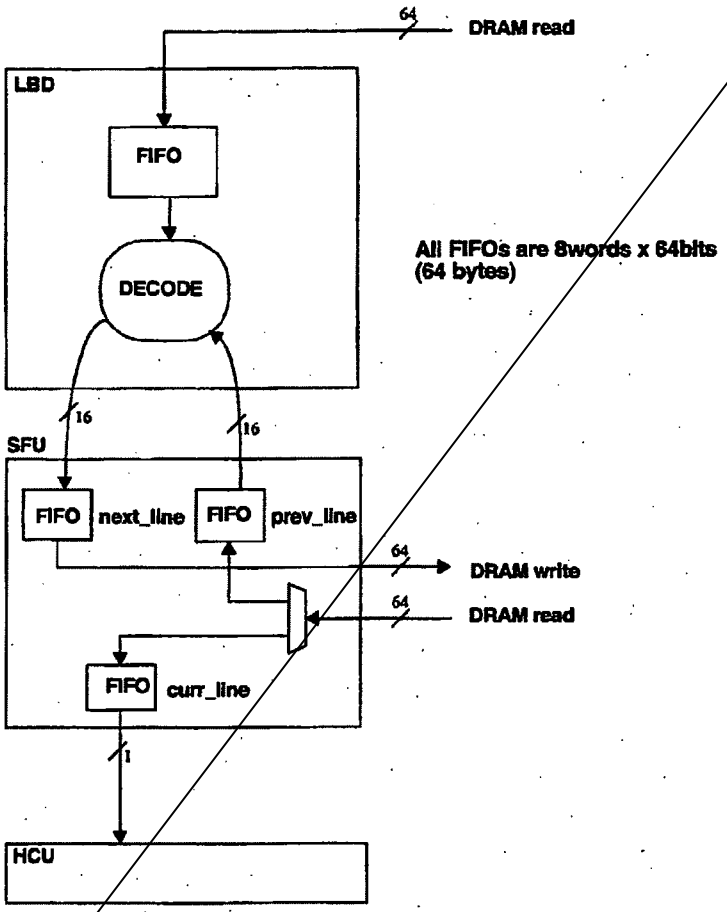

64
DRAM read
LBD
FIFO
DECODE
All FIFOs are 8words x 64bits
(64 bytes)
16
16
SFU
FIFO
next_line
FIFO
prev_line
64
DRAM write
64
DRAM read
FIFO
curr_line
1
HCU

FIG. 160

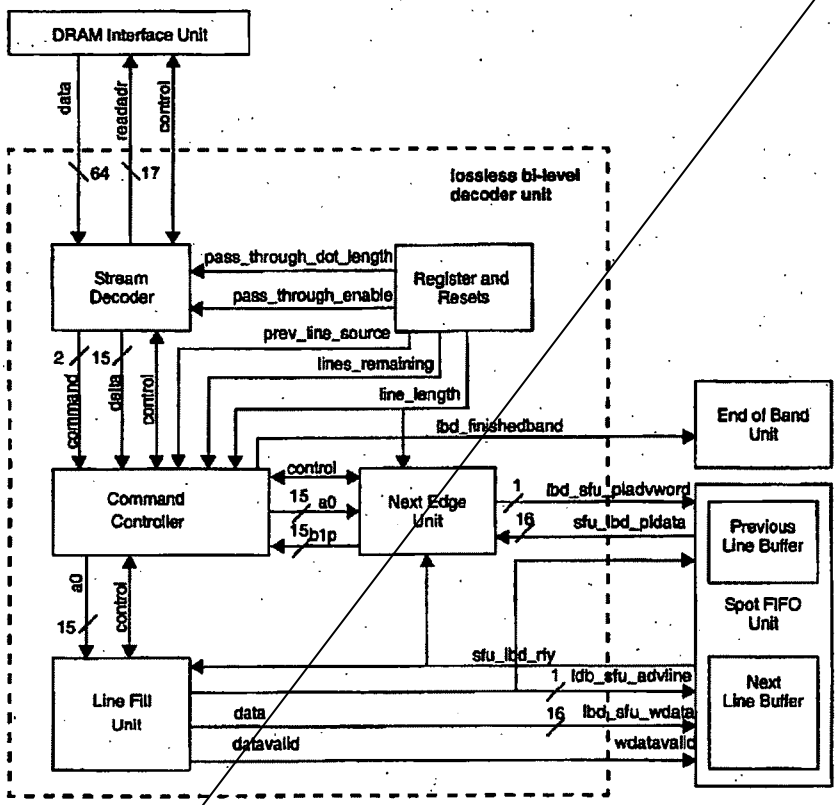

DRAM Interface Unit
data
readadr
control
64
17
lossless bi-level decoder unit
Stream Decoder
pass_through_dot_length
pass_through_enable
Register and Resets
prev_line_source
lines_remaining
line_length
2
15
command
data
control
lbd_finishedband
End of Band Unit
Command Controller
control
15 a0
15 b1p
Next Edge Unit
1
lbd_sfu_pladvword
16
sfu_lbd_pldata
Previous Line Buffer
Spot FIFO Unit
a0
15
control
sfu_lbd_rfy
1
ldb_sfu_advline
Next Line Buffer
Line Fill Unit
data
16
lbd_sfu_wdata
datavalid
wdatavalid

FIG. 161

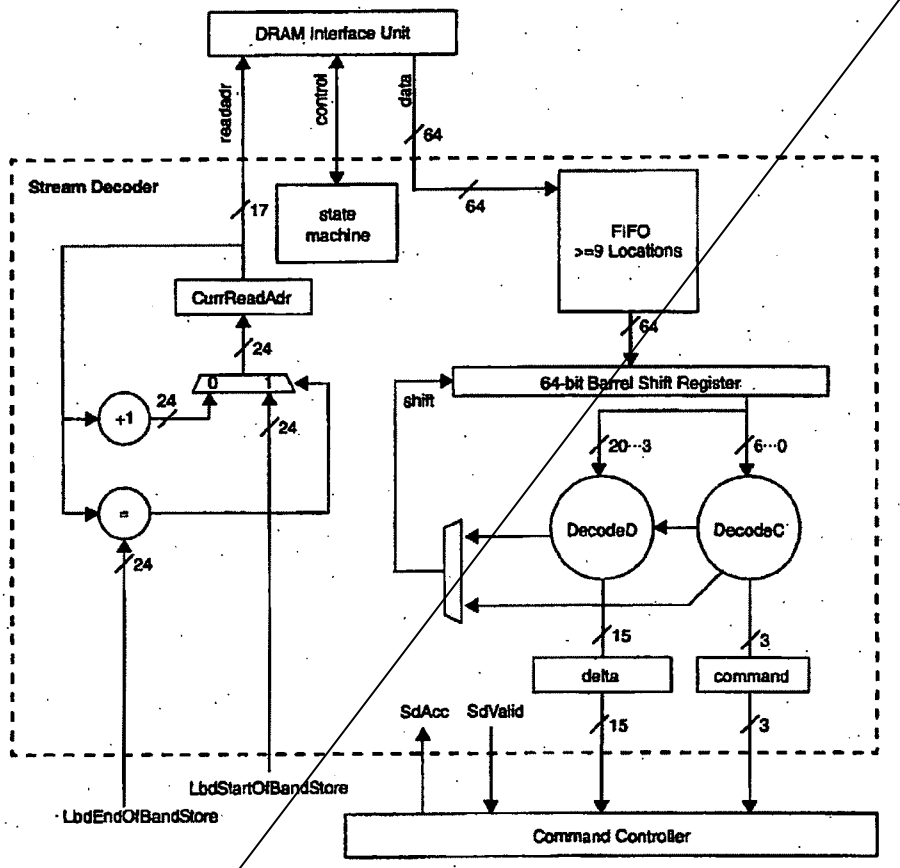

DRAM Interface Unit
readadr
control
data
64
Stream Decoder
17
state machine
64
FIFO >=9 Locations
CurrReadAdr
24
64
64-bit Barrel Shift Register
shift
0
1
24
24
+1
20···3
6···0
DecodeD
DecodeC
=
24
15
3
delta
command
SdAcc
SdValid
15
3
LbdStartOfBandStore
LbdEndOfBandStore
Command Controller

FIG. 162

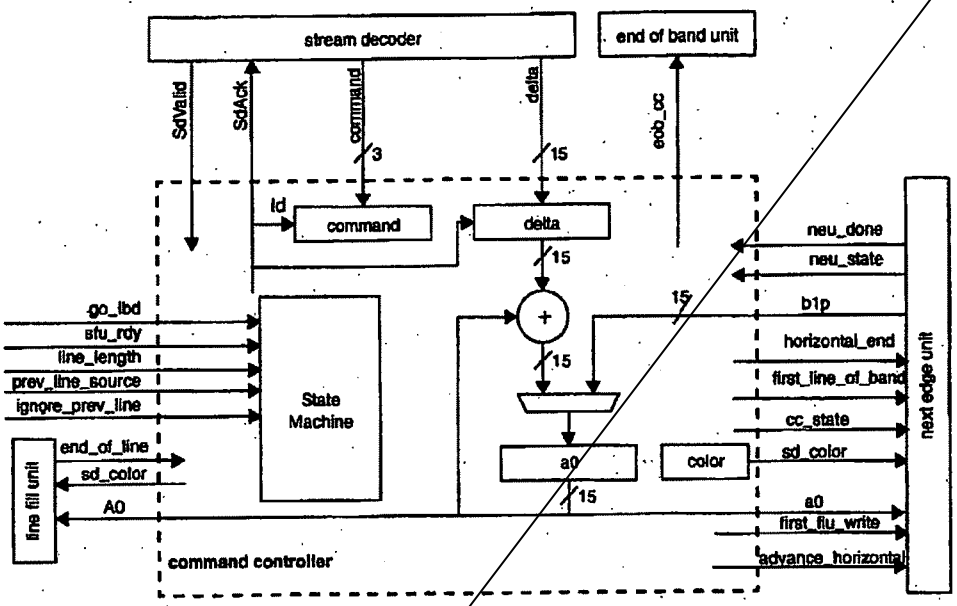


FIG. 163

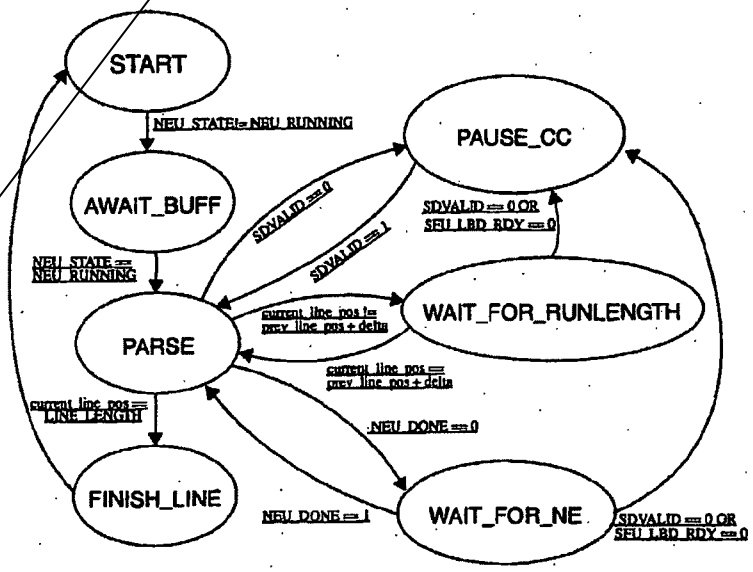


FIG. 164

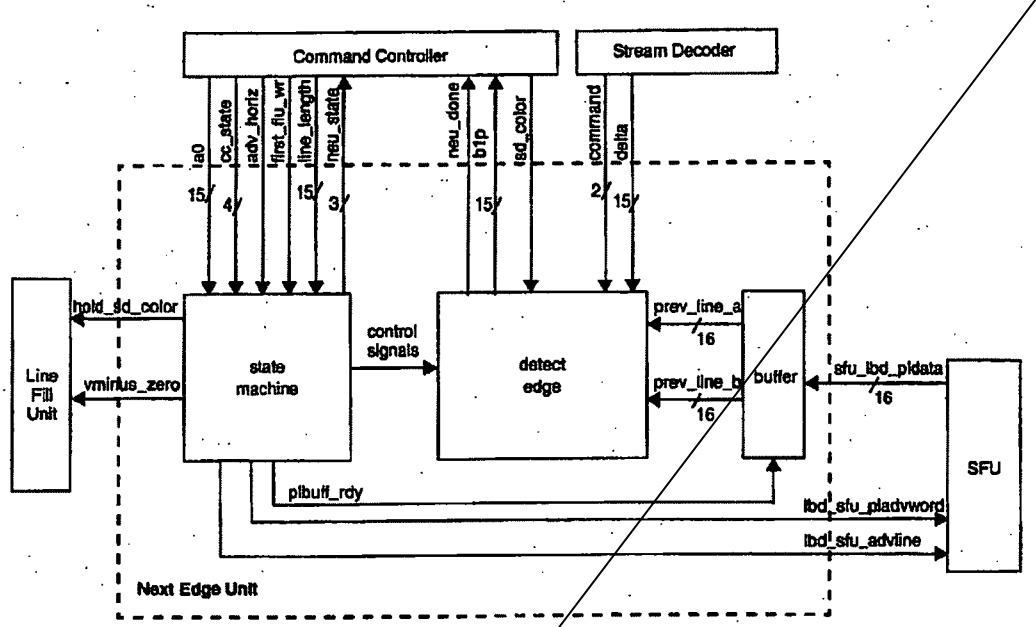


FIG. 165

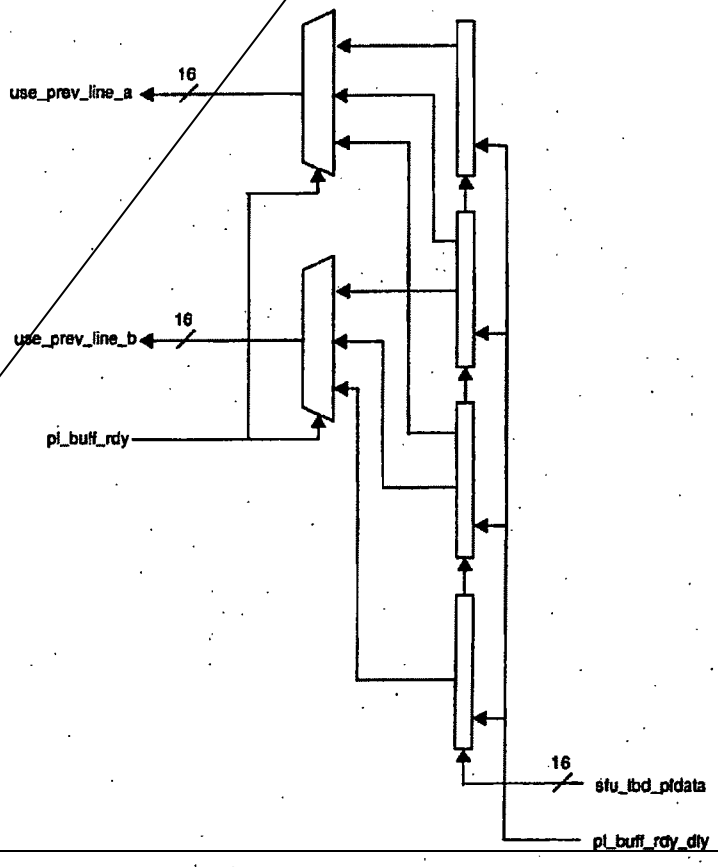


FIG. 166

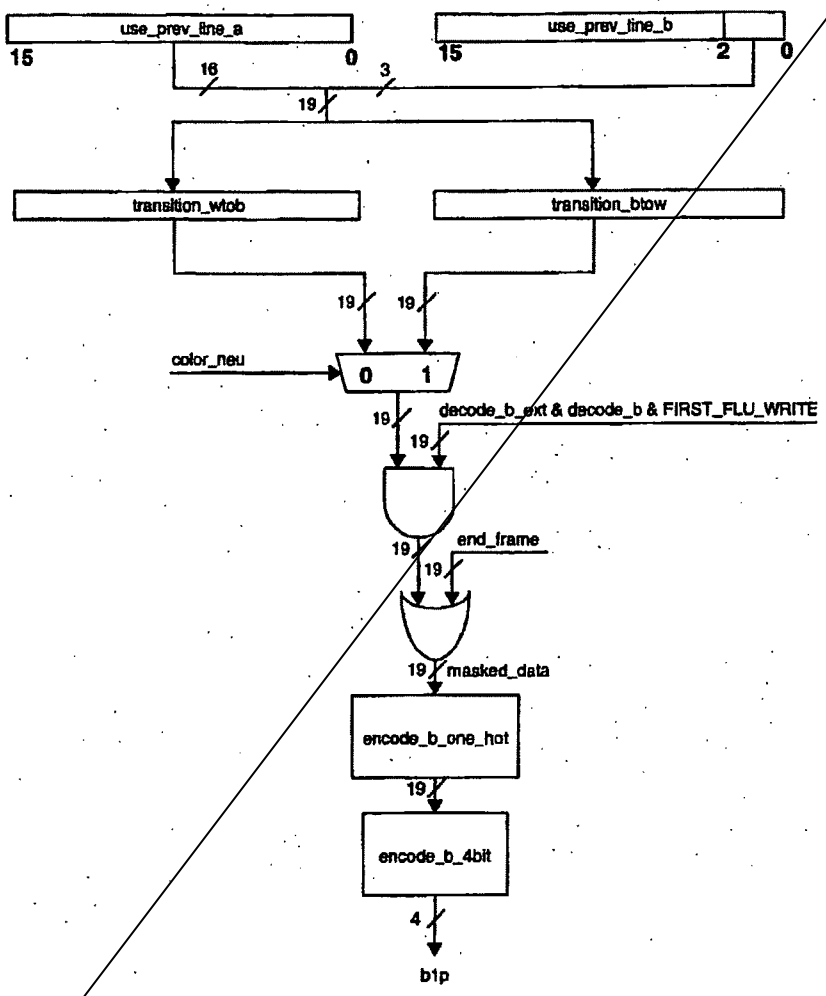

use_prev_line_a
15
0
use_prev_line_b
15
2
0
16
3
19
transition_wtob
transition_btow
19
19
color_neu
0
1
19
decode_b_ext & decode_b & FIRST_FLU_WRITE
19
19
end_frame
19
19
masked_data
encode_b_one_hot
19
encode_b_4bit
4
b1p

FIG. 167

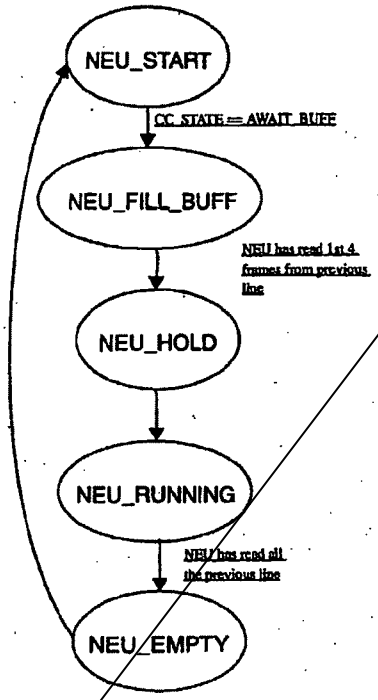

NEU_START
CC_STATE == AWAIT_BUFF
NEU_FILL_BUFF
NEU has read 1st 4 frames from previous line
NEU_HOLD
NEU_RUNNING
NEU has read all the previous line
NEU_EMPTY

FIG. 168

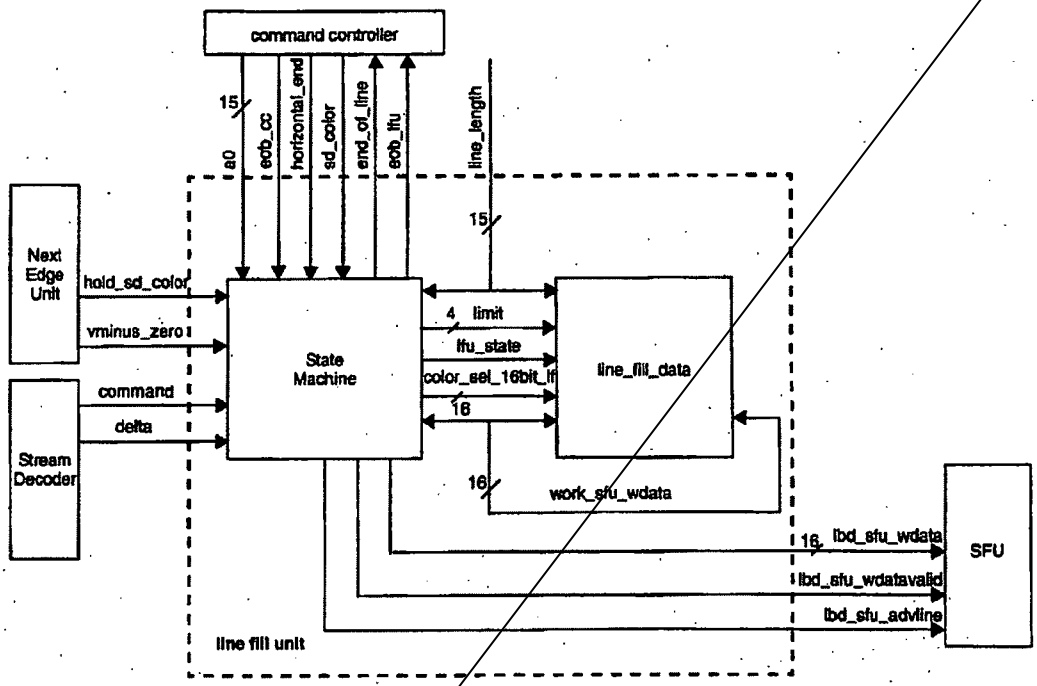


FIG. 169

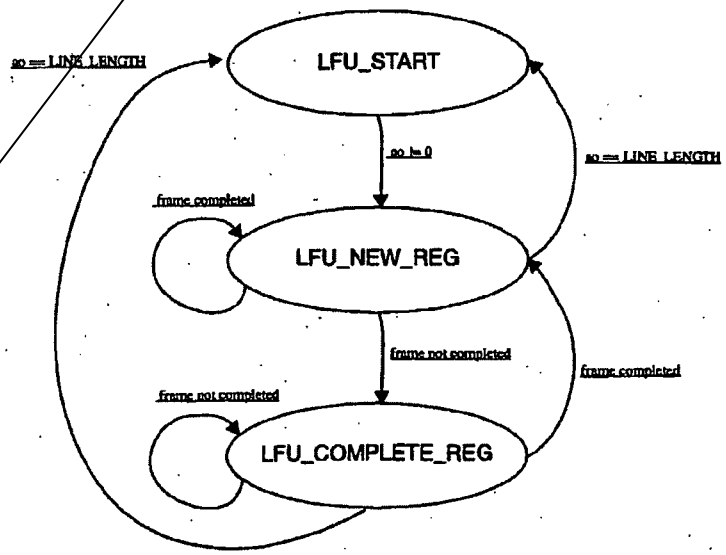


FIG. 170

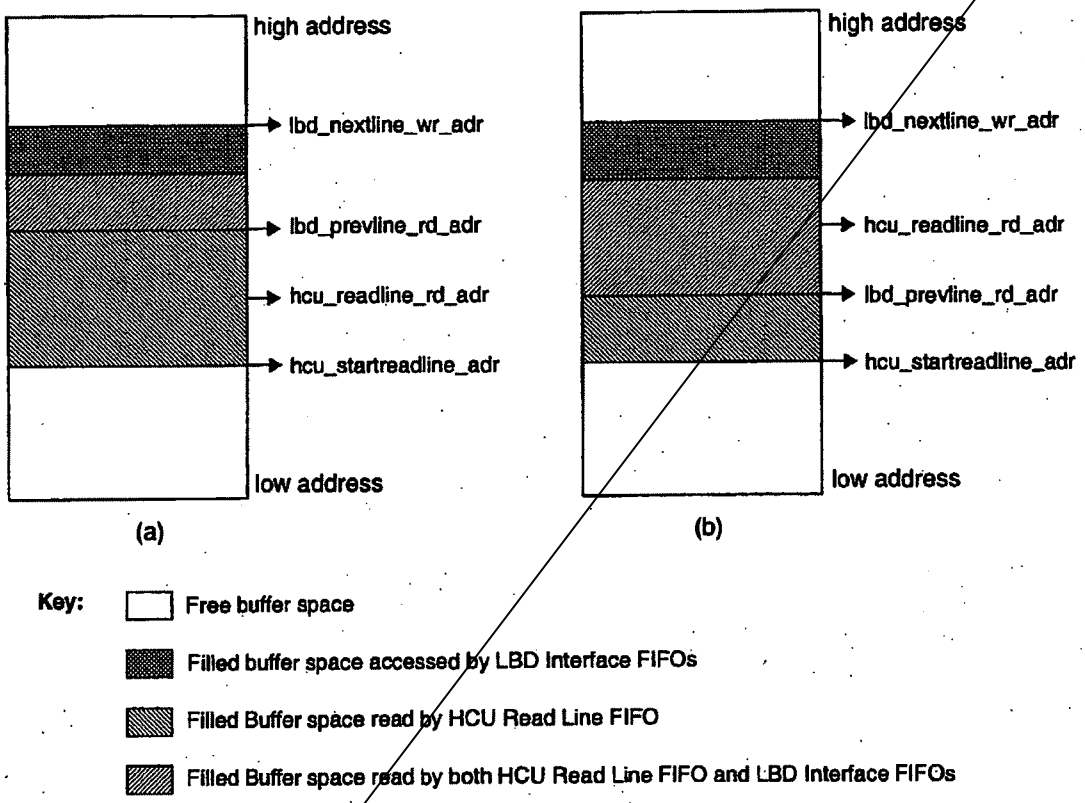

high address
lbd_nextline_wr_adr
lbd_previine_rd_adr
hcu_readline_rd_adr
hcu_startreadline_adr
low address
(a)
high address
lbd_nextline_wr_adr
hcu_readline_rd_adr
lbd_previine_rd_adr
hcu_startreadline_adr
low address
(b)
Key:
Free buffer space
Filled buffer space accessed by LBD Interface FIFOs
Filled Buffer space read by HCU Read Line FIFO
Filled Buffer space read by both HCU Read Line FIFO and LBD Interface FIFOs

FIG. 171

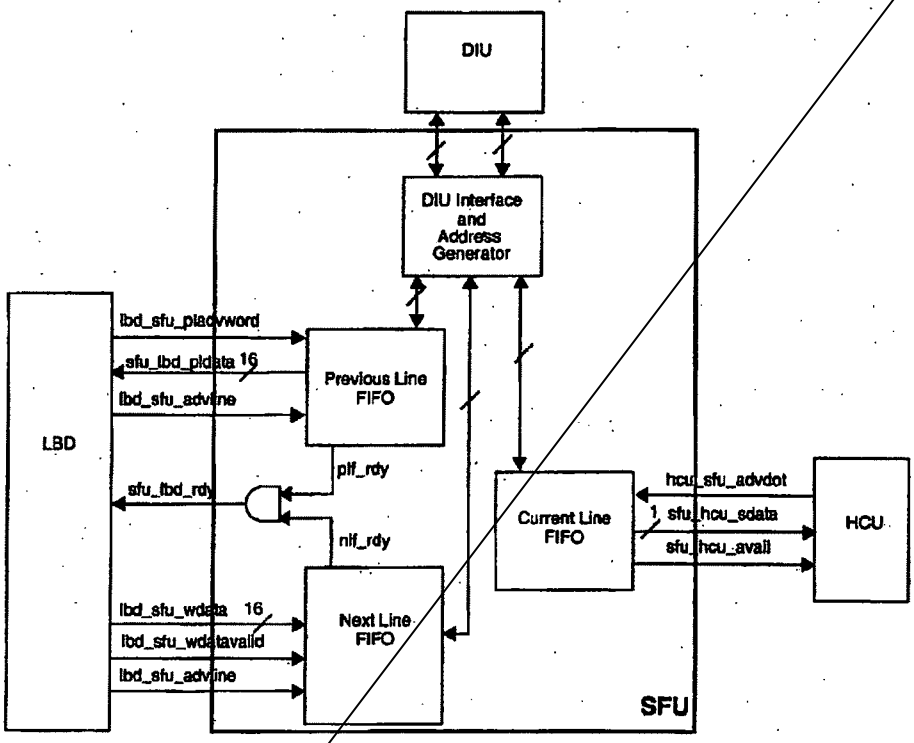

DIU
DIU Interface and Address Generator
LBD
lbd_sfu_pladvword
sfu_lbd_pldata 16
lbd_sfu_advline
Previous Line FIFO
plf_rdy
sfu_lbd_rdy
nlf_rdy
Current Line FIFO
hcu_sfu_advdot
1 sfu_hcu_sdata
sfu_hcu_avail
HCU
lbd_sfu_wdata 16
lbd_sfu_wdatavalid
lbd_sfu_advline
Next Line FIFO
SFU

FIG. 172

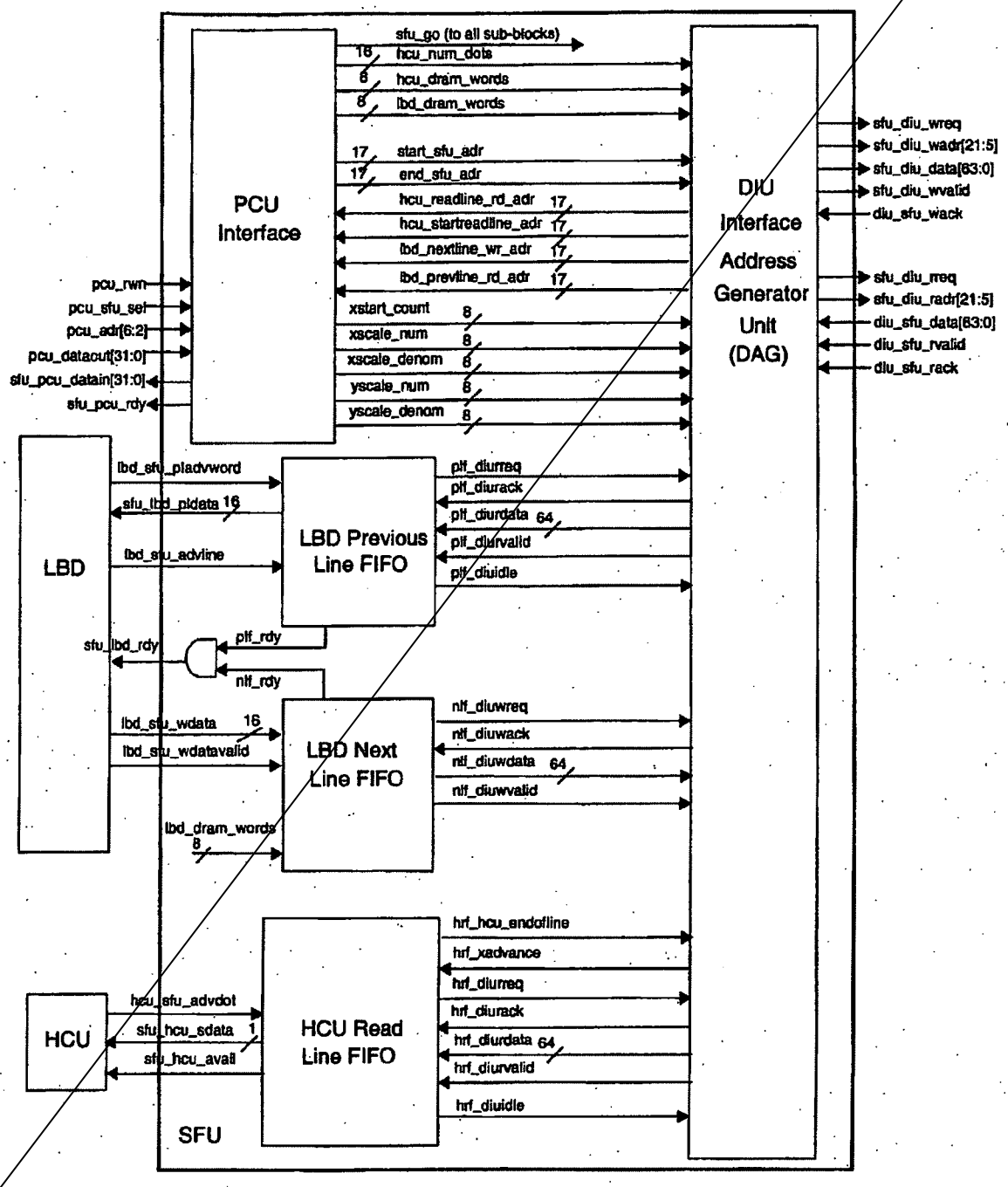

PCU Interface
DIU Interface Address Generator Unit (DAG)
LBD
LBD Previous Line FIFO
LBD Next Line FIFO
HCU
HCU Read Line FIFO
SFU
sfu_go (to all sub-blocks)
hcu_num_dots
hcu_dram_words
lbd_dram_words
start_sfu_adr
end_sfu_adr
hcu_readline_rd_adr
hcu_startreadline_adr
lbd_nextline_wr_adr
lbd_prevline_rd_adr
xstart_count
xscale_num
xscale_denom
yscale_num
yscale_denom
pcu_rwn
pcu_sfu_sel
pcu_adr[6:2]
pcu_datacut[31:0]
sfu_pcu_datain[31:0]
sfu_pcu_rdy
sfu_diu_wreq
sfu_diu_wadr[21:5]
sfu_diu_data[63:0]
sfu_diu_wvalid
diu_sfu_wack
sfu_diu_rreq
sfu_diu_radr[21:5]
diu_sfu_data[63:0]
diu_sfu_rvalid
diu_sfu_rack
lbd_sfu_pladvword
sfu_lbd_pldata
lbd_sfu_advline
sfu_lbd_rdy
plf_rdy
nlf_rdy
plf_diureq
plf_diurack
plf_diurdata
plf_diurvalid
plf_diuidle
lbd_sfu_wdata
lbd_sfu_wdatavalid
lbd_dram_words
nlf_diuwreq
nlf_diuwack
nlf_diuwdata
nlf_diuwvalid
hcu_sfu_advdot
sfu_hcu_sdata
sfu_hcu_avail
hrf_hcu_endofline
hrf_xadvance
hrf_diureq
hrf_diurack
hrf_diurdata
hrf_diurvalid
hrf_diuidle

FIG. 173

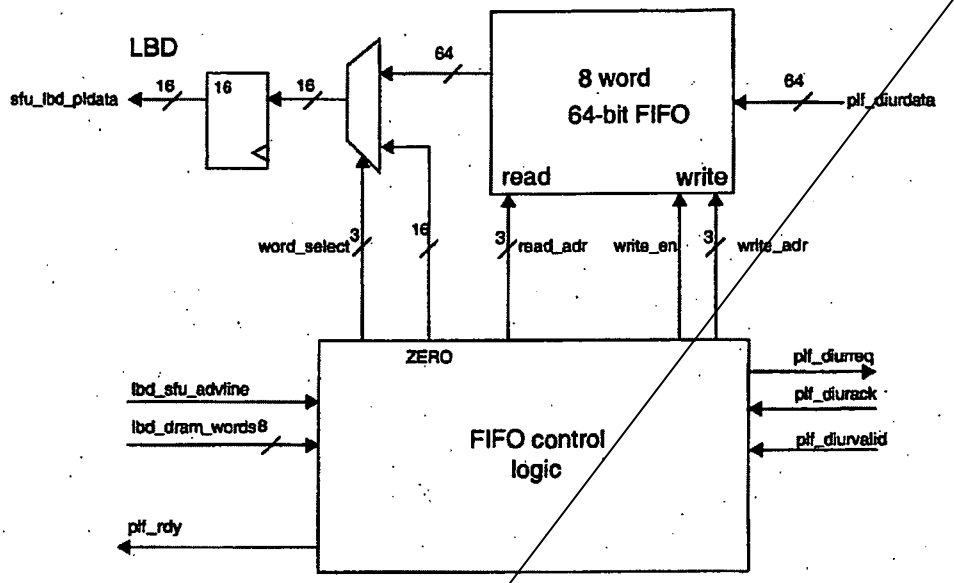


FIG. 174

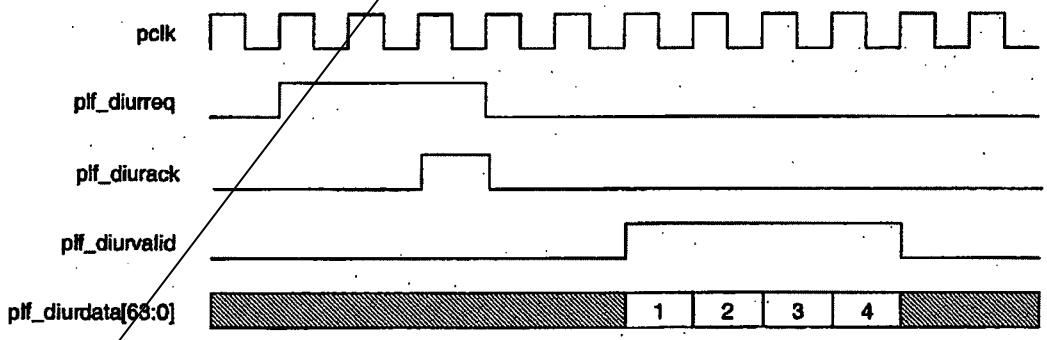


FIG. 175

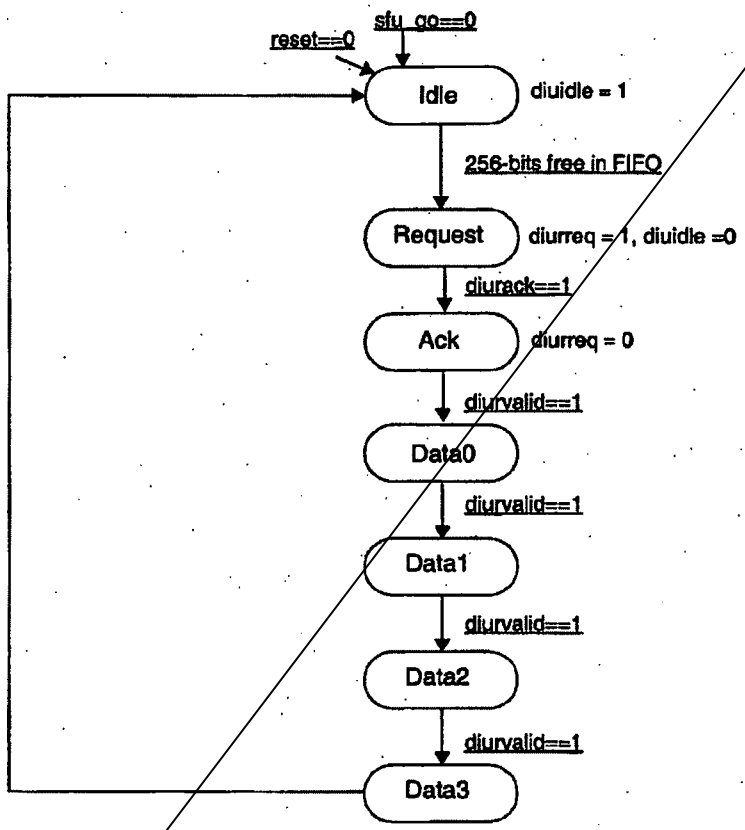

sfu_go==0
reset==0
Idle
diuidle = 1
256-bits free in FIFO
Request
diurreq = 1, diuidle =0
diurack==1
Ack
diurreq = 0
diurvalid==1
Data0
diurvalid==1
Data1
diurvalid==1
Data2
diurvalid==1
Data3

FIG. 176

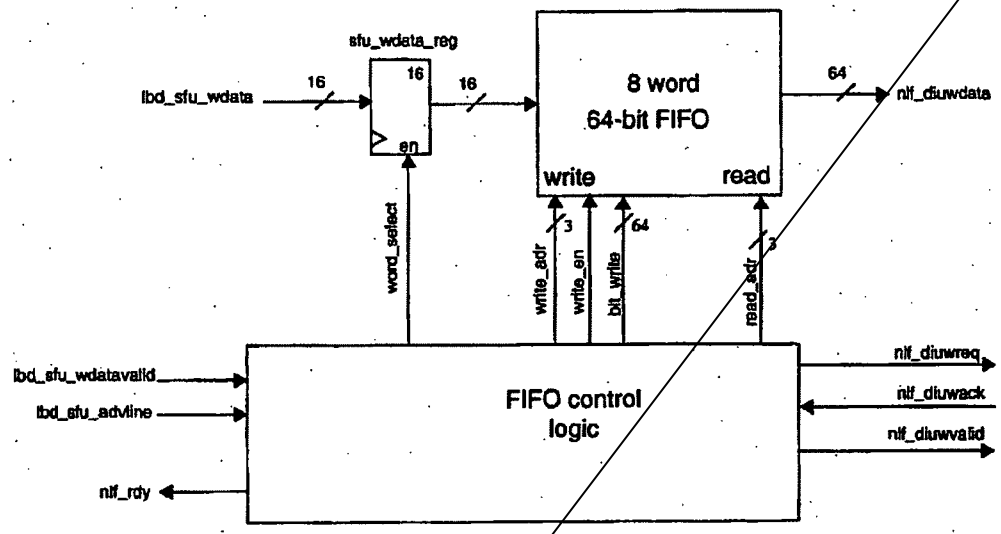


FIG. 177

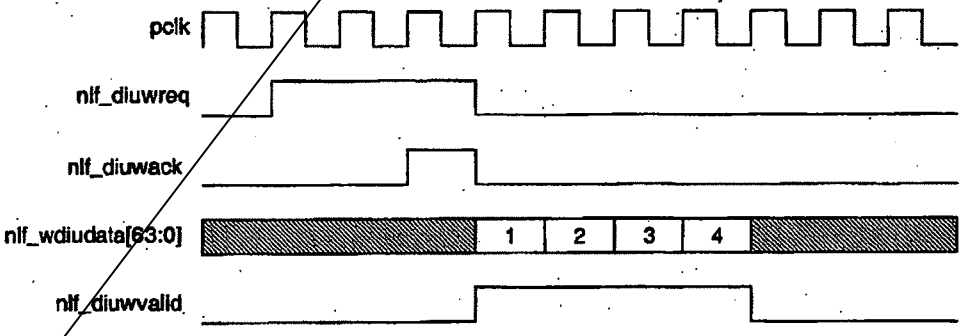


FIG. 178

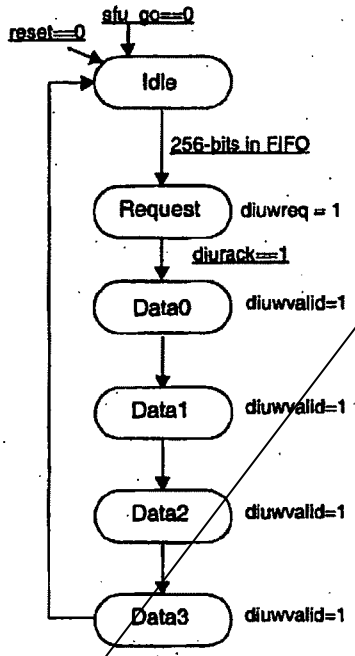


FIG. 179

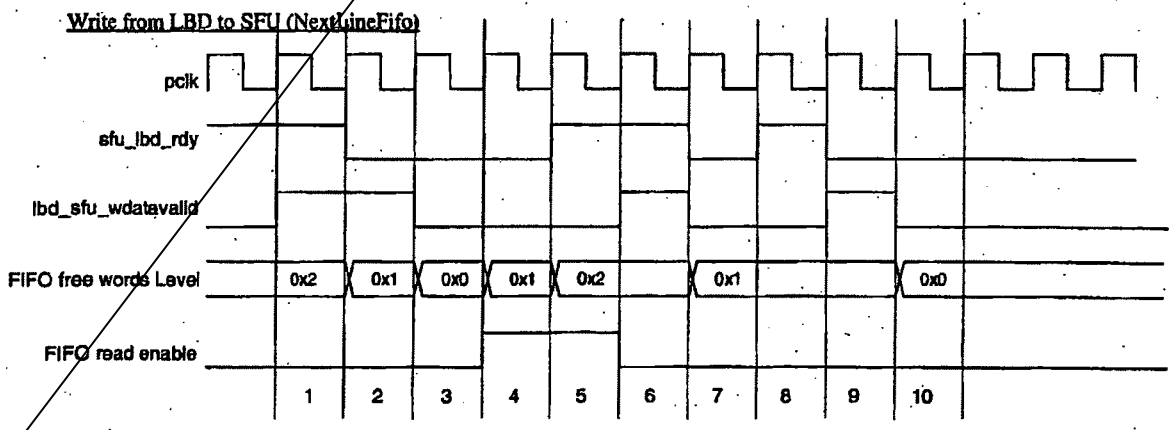


FIG. 180

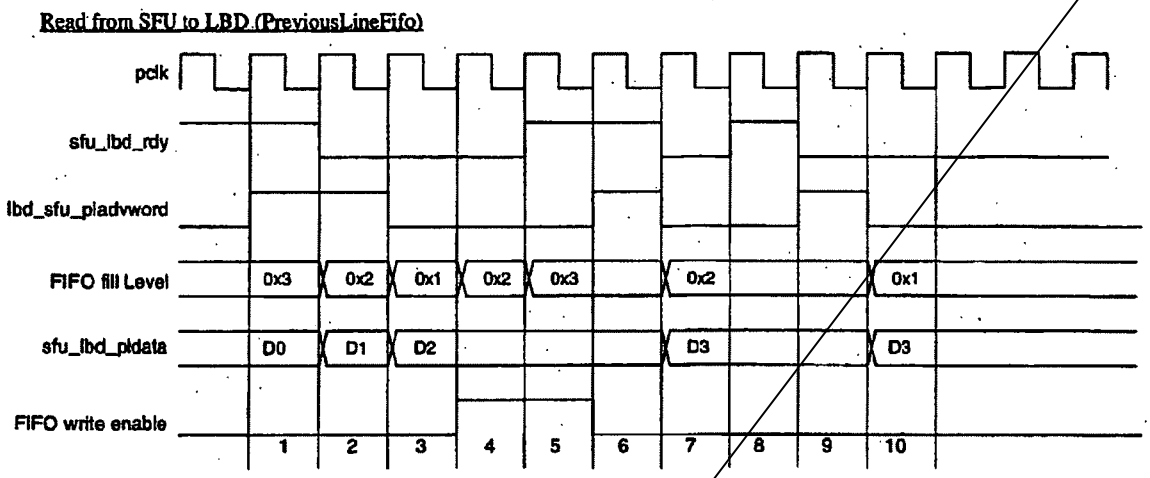


FIG. 181

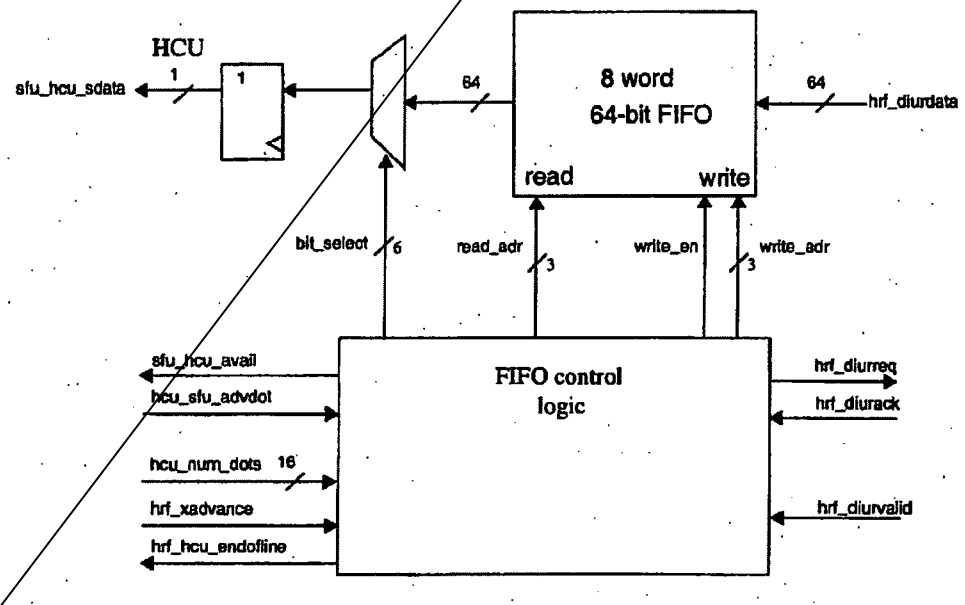


FIG. 182

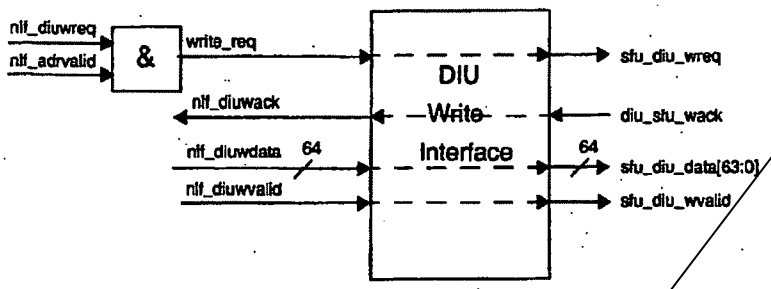


FIG. 183

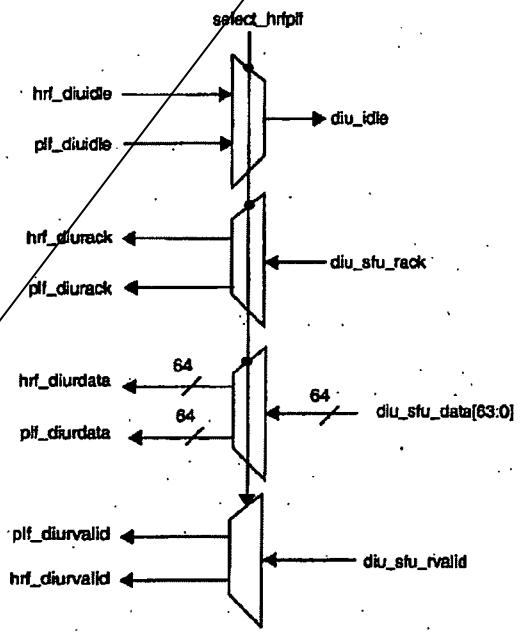


FIG. 184

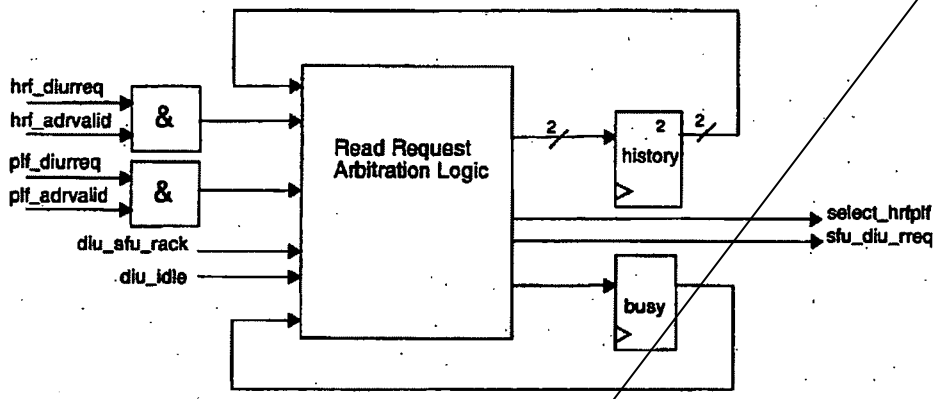


FIG. 185

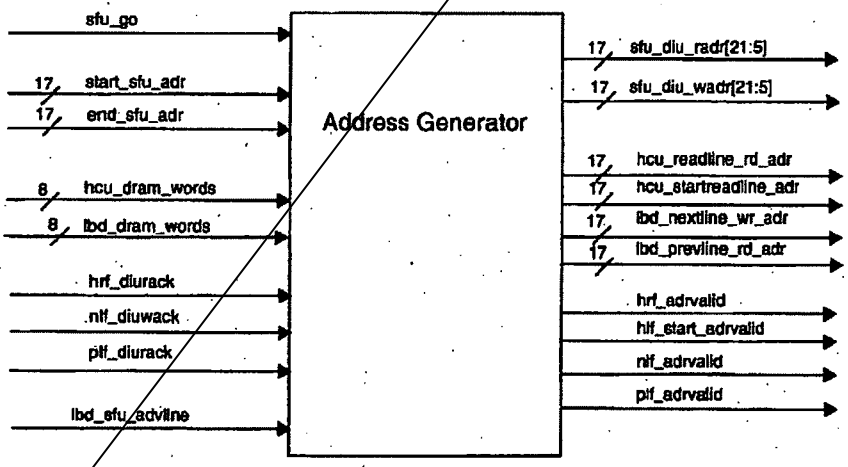


FIG. 186

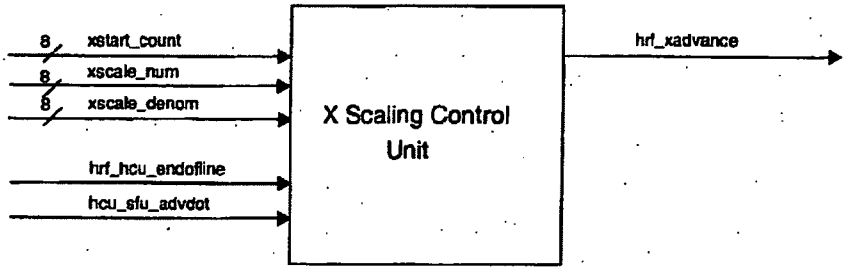


FIG. 187

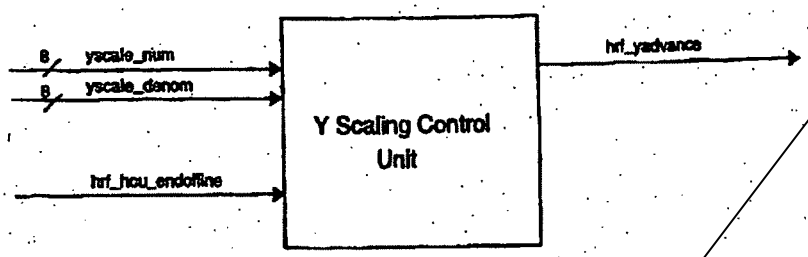


FIG. 188

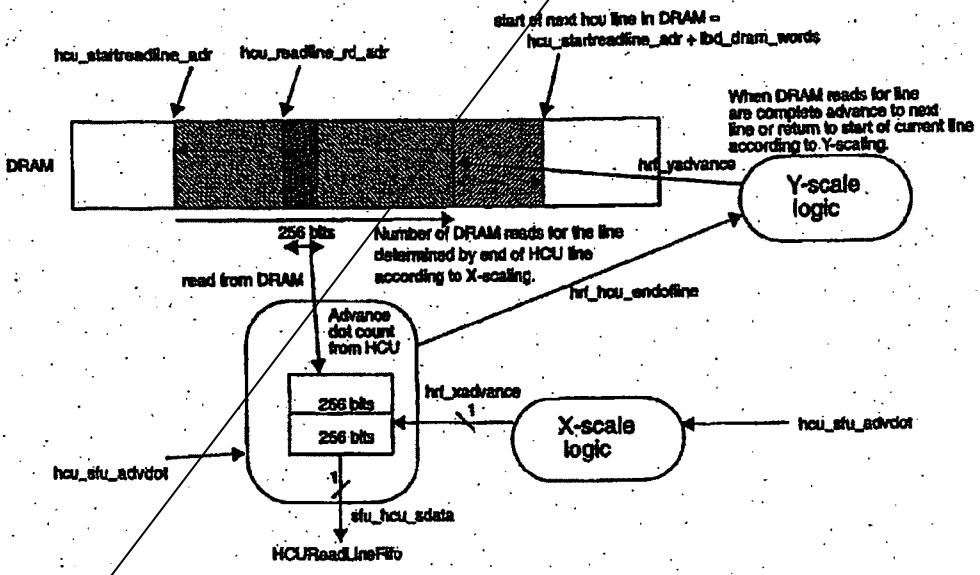


FIG. 189

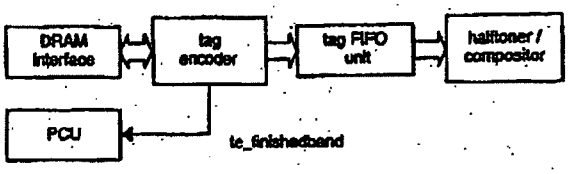


FIG. 190

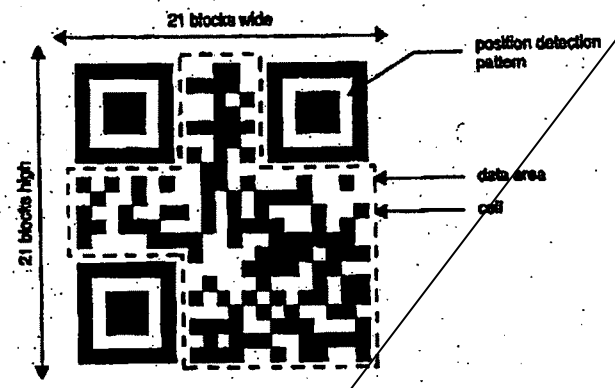


FIG. 191

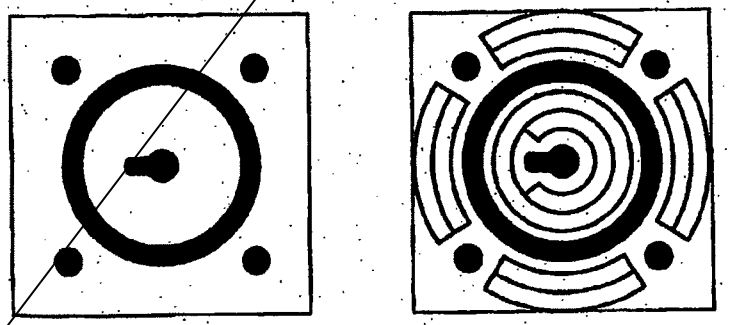


FIG. 192

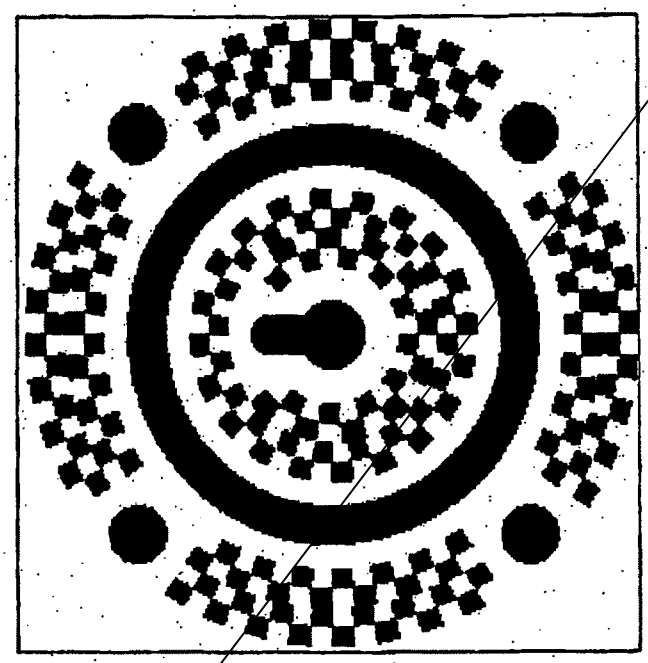


FIG. 193

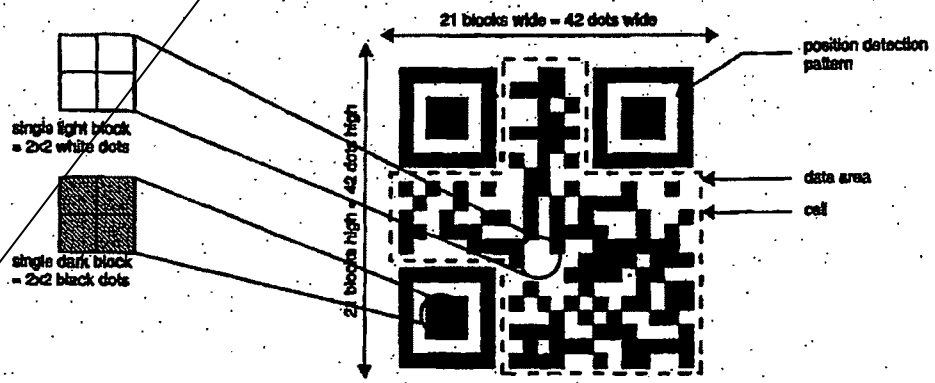


FIG. 194

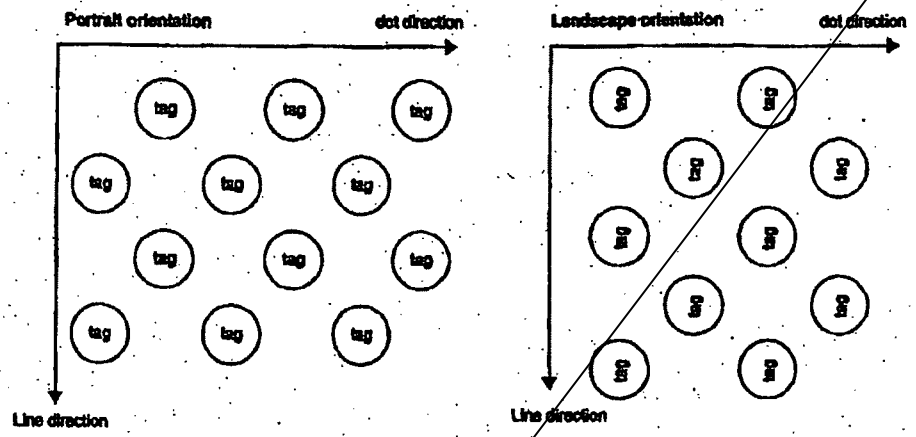


FIG. 195

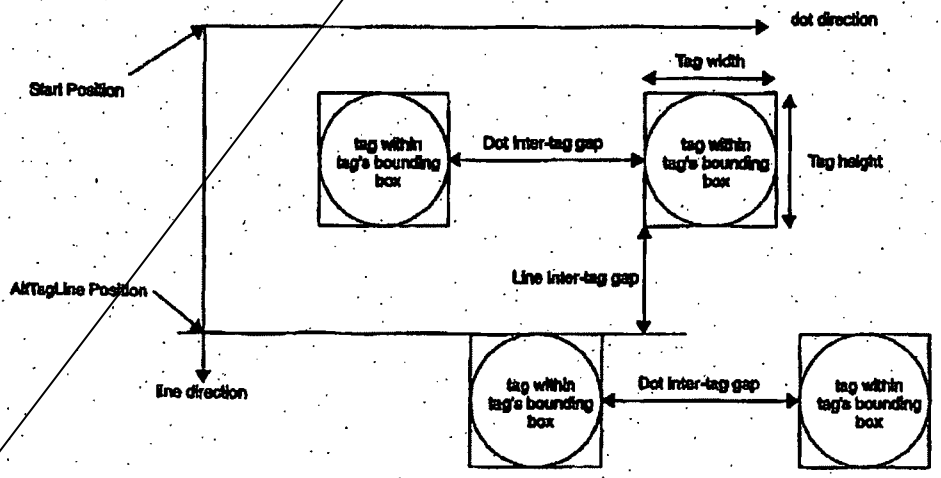


FIG. 196

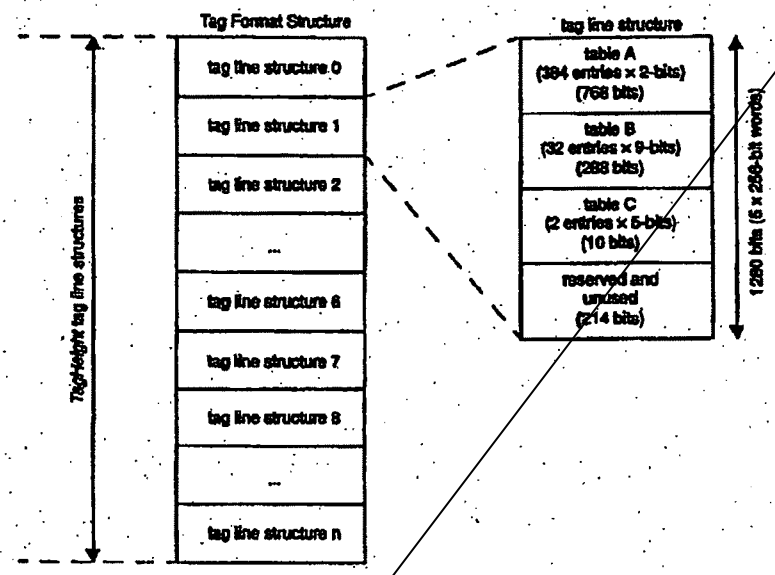


FIG. 197

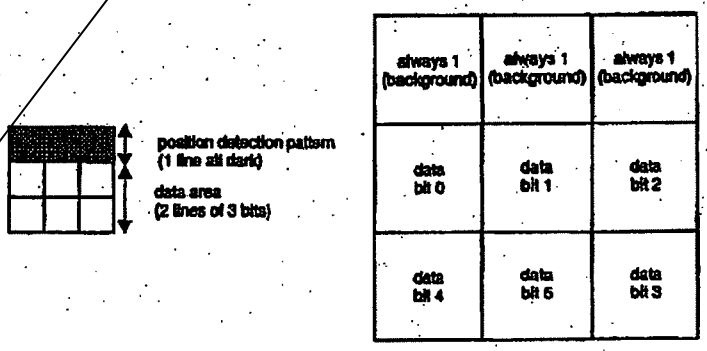


FIG. 198

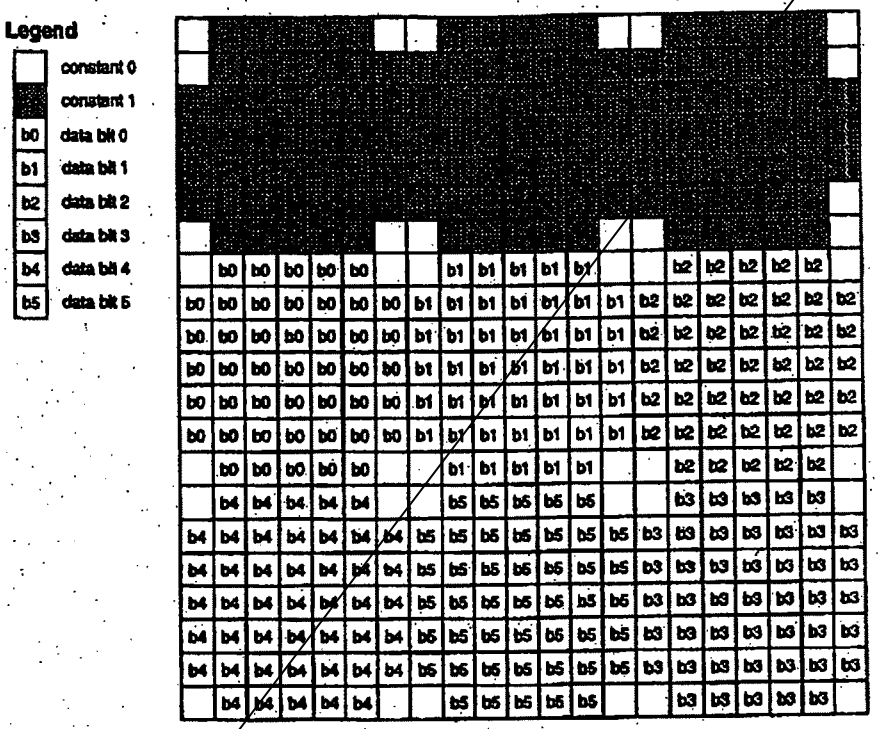

Legend
constant 0
constant 1
b0 data bit 0
b1 data bit 1
b2 data bit 2
b3 data bit 3
b4 data bit 4
b5 data bit 5

FIG. 199

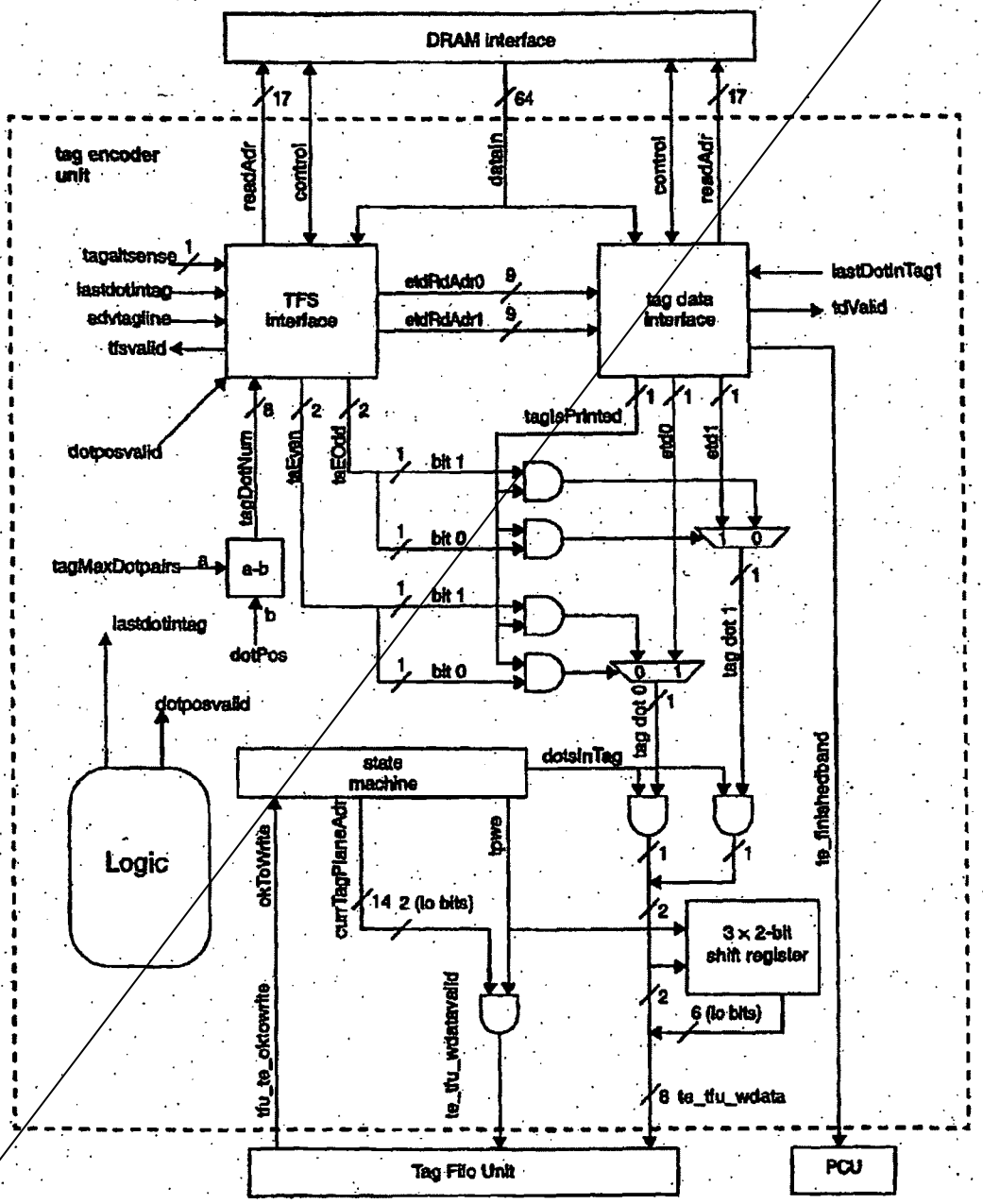

DRAM interface
17
64
17
tag encoder unit
readAdr
control
datain
control
readAdr
tagaltsense
lastdotintag
advtagline
tfsvalid
TFS interface
etdRdAdr0
etdRdAdr1
tag data interface
lastDotInTag1
tdValid
dotposvalid
tagDotNum
teEven
teEOdd
tagIsPrinted
etd0
etd1
bit 1
bit 0
tagMaxDotpairs
a-b
b
dotPos
lastdotintag
dotposvalid
tag dot 1
tag dot 0
state machine
dotsInTag
Logic
okToWrite
currTagPlaneAdr
tpwe
14
2 (lo bits)
3 × 2-bit shift register
6 (lo bits)
te_finishedband
tfu_te_oktowrite
te_tfu_wdatavalid
8
te_tfu_wdata
Tag Fifo Unit
PCU

FIG. 200

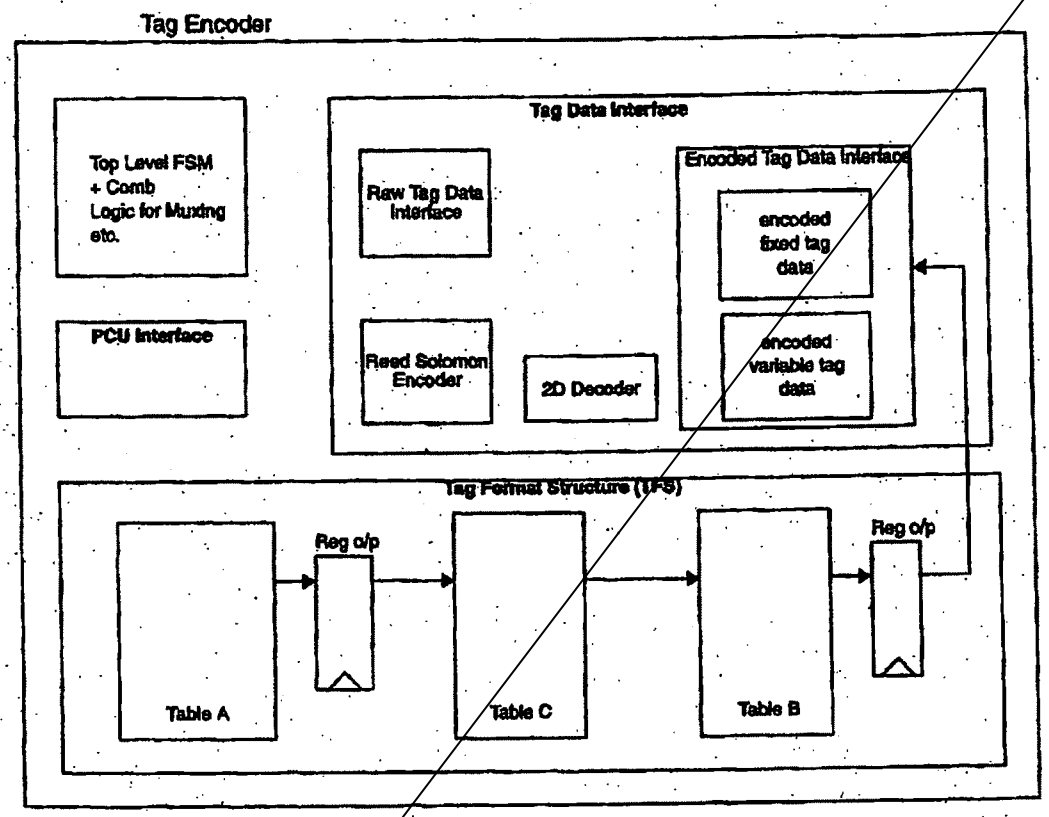


FIG. 201

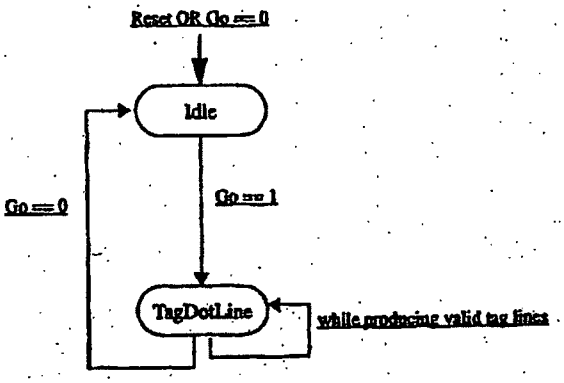


FIG. 202

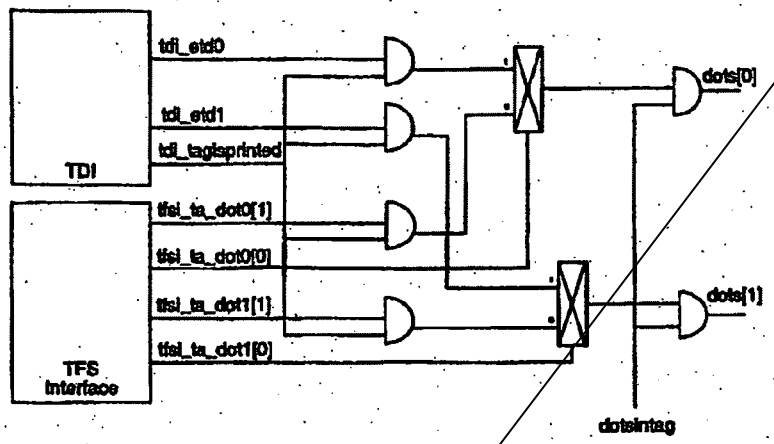


FIG. 203

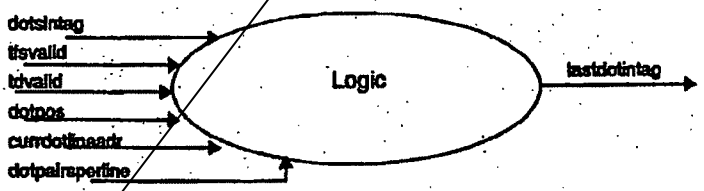


FIG. 204

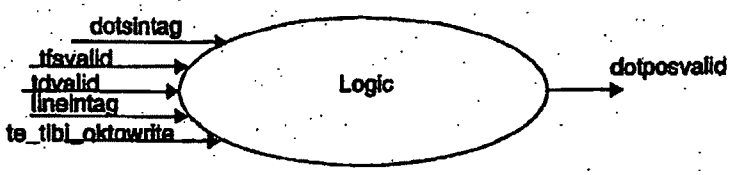


FIG. 205

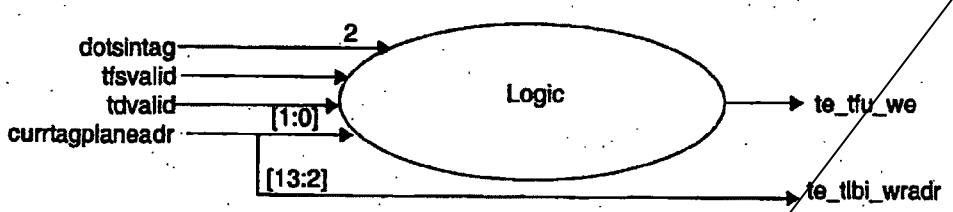


FIG. 206

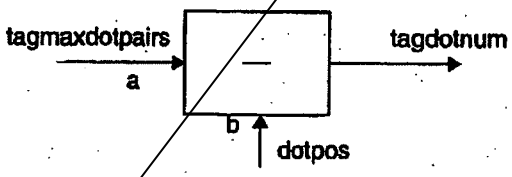


FIG. 207

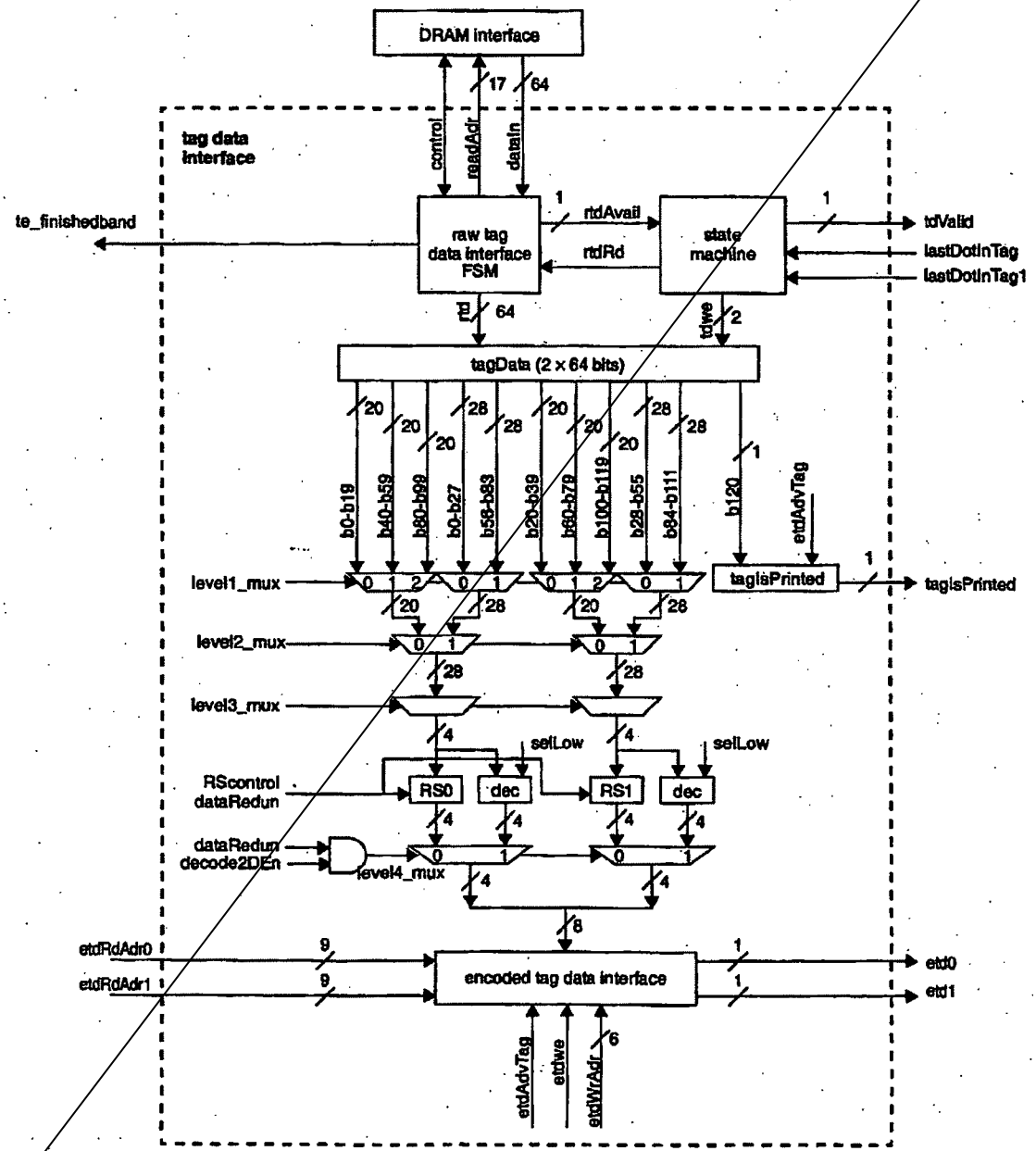

DRAM interface
17
64
tag data interface
control
readAdr
dataIn
te_finishedband
raw tag data interface FSM
rtdAvail
rtdRd
state machine
tdValid
lastDotInTag
lastDotInTag1
64
tdwe
2
tagData (2 x 64 bits)
b0-b19
b40-b59
b80-b99
b0-b27
b58-b83
b20-b39
b60-b79
b100-b119
b28-b55
b84-b111
b120
etdAdrTag
tagIsPrinted
tagIsPrinted
level1_mux
level2_mux
level3_mux
selLow
selLow
RScontrol
dataRedun
RS0
dec
RS1
dec
dataRedun
decode2DEn
level4_mux
etdRdAdr0
etdRdAdr1
encoded tag data interface
etd0
etd1
etdAdrTag
etdwe
etdWrAdr
6

FIG. 208

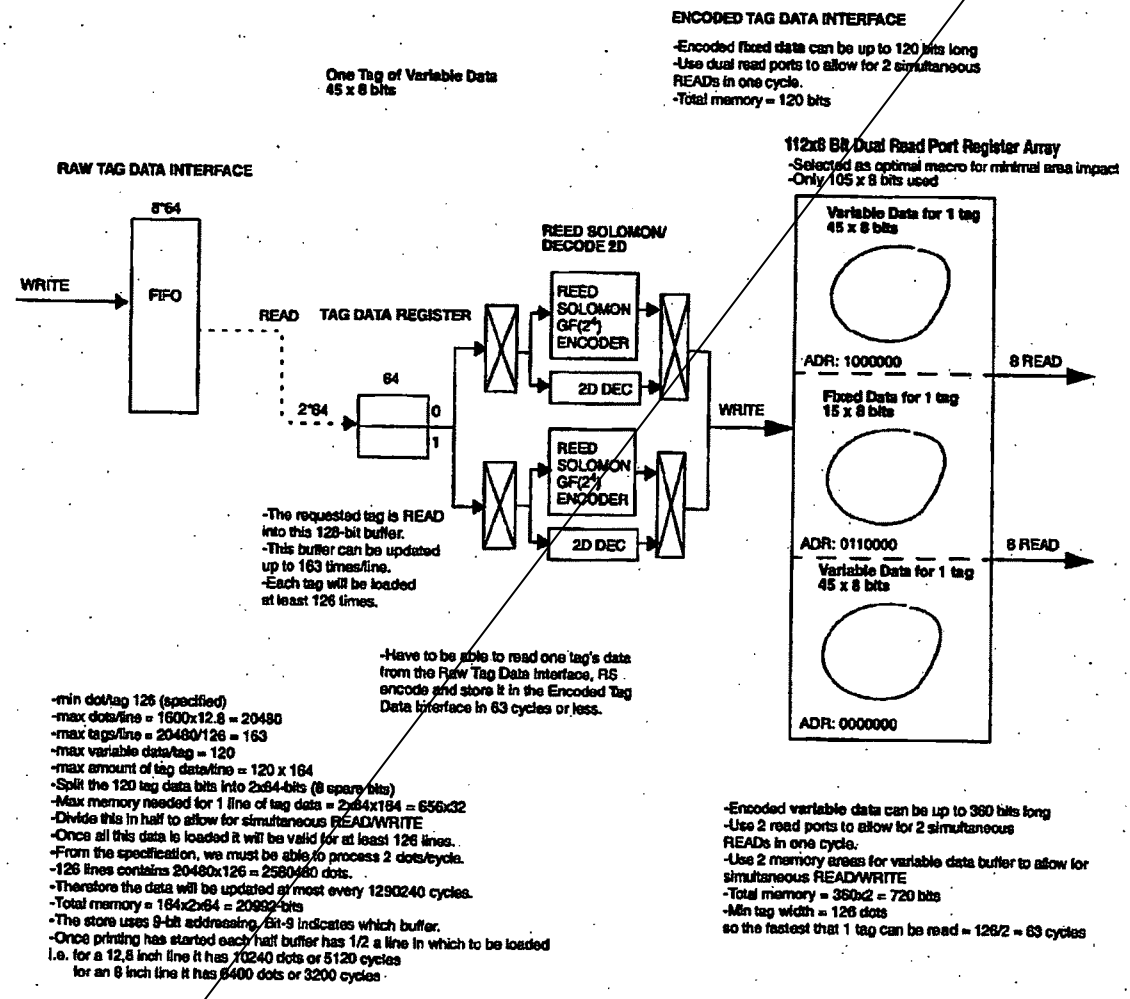

ENCODED TAG DATA INTERFACE
-Encoded fixed data can be up to 120 bits long
-Use dual read ports to allow for 2 simultaneous READs in one cycle.
-Total memory = 120 bits
One Tag of Variable Data
45 x 8 bits
112x8 8R Dual Read Port Register Array
-Selected as optimal macro for minimal area impact
-Only 105 x 8 bits used
RAW TAG DATA INTERFACE
8*64
WRITE
FIFO
READ
TAG DATA REGISTER
2*64
64
0
1
REED SOLOMON/ DECODE 2D
REED SOLOMON GF(2^8) ENCODER
2D DEC
REED SOLOMON GF(2^8) ENCODER
2D DEC
WRITE
Variable Data for 1 tag
45 x 8 bits
ADR: 1000000
8 READ
Fixed Data for 1 tag
15 x 8 bits
ADR: 0110000
8 READ
Variable Data for 1 tag
45 x 8 bits
ADR: 0000000
-The requested tag is READ into this 128-bit buffer.
-This buffer can be updated up to 163 times/line.
-Each tag will be loaded at least 126 times.
-Have to be able to read one tag's data from the Raw Tag Data Interface, RS encode and store it in the Encoded Tag Data Interface in 63 cycles or less.
-min dot/tag 126 (specified)
-max dots/line = 1600x12.8 = 20480
-max tags/line = 20480/126 = 163
-max variable data/tag = 120
-max amount of tag data/line = 120 x 164
-Split the 120 tag data bits into 2x64-bits (8 spare bits)
-Max memory needed for 1 line of tag data = 2x64x164 = 656x32
-Divide this in half to allow for simultaneous READ/WRITE
-Once all this data is loaded it will be valid for at least 126 lines.
-From the specification, we must be able to process 2 dots/cycle.
-126 lines contains 20480x126 = 2580480 dots.
-Therefore the data will be updated at most every 1290240 cycles.
-Total memory = 164x2x64 = 20992 bits
-The store uses 9-bit addressing. Bit-9 indicates which buffer.
-Once printing has started each half buffer has 1/2 a line in which to be loaded
I.e. for a 12.8 inch line it has 10240 dots or 5120 cycles
for an 8 inch line it has 6400 dots or 3200 cycles
-Encoded variable data can be up to 360 bits long
-Use 2 read ports to allow for 2 simultaneous READs in one cycle.
-Use 2 memory areas for variable data buffer to allow for simultaneous READ/WRITE
-Total memory = 360x2 = 720 bits
-Min tag width = 126 dots
so the fastest that 1 tag can be read = 126/2 = 63 cycles

FIG. 209

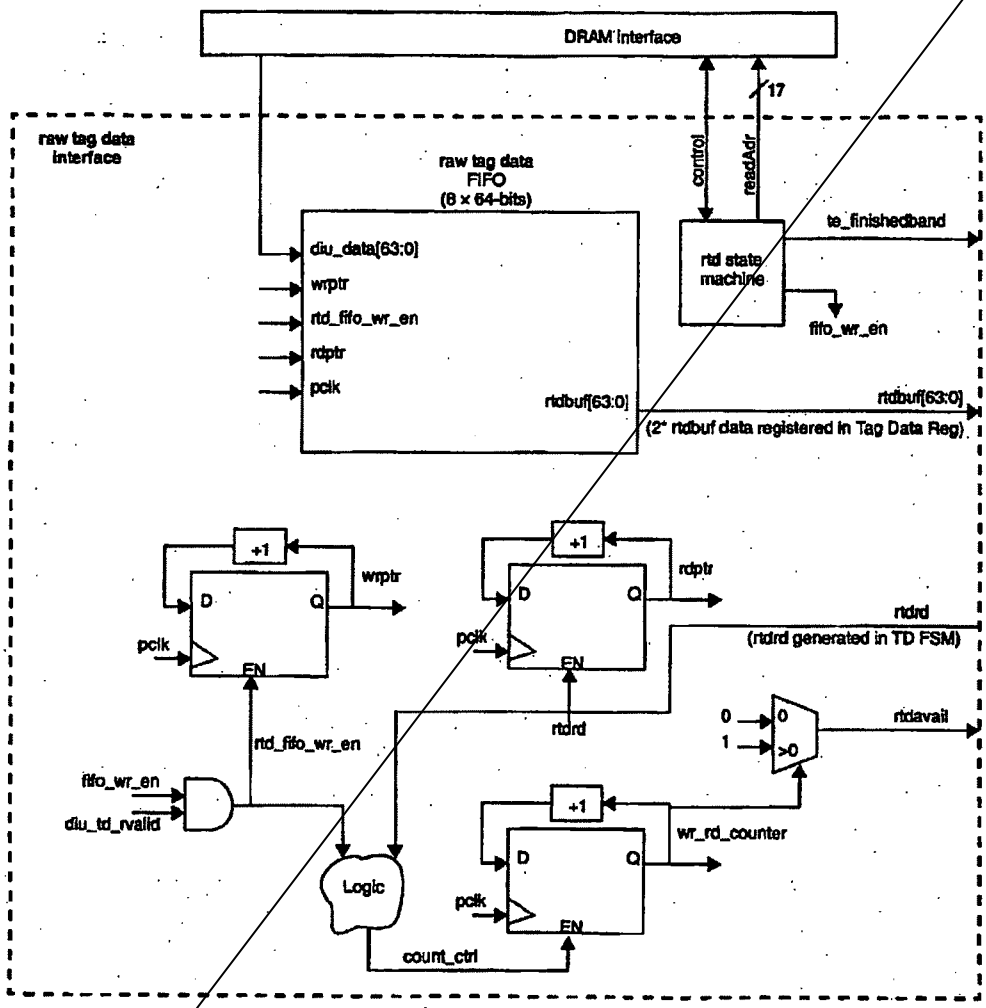

DRAM Interface
17
control
readAdr
raw tag data interface
raw tag data FIFO (8 x 64-bits)
diu_data[63:0]
wrptr
rtd_fifo_wr_en
rdptr
pclk
rtdbuf[63:0]
rtd state machine
te_finishedband
fifo_wr_en
rtdbuf[63:0]
(2* rtdbuf data registered in Tag Data Reg)
+1
D
Q
EN
wrptr
rdptr
rtdrd
(rtdrd generated in TD FSM)
rtdavail
0
1
>0
rtd_fifo_wr_en
fifo_wr_en
diu_td_rvalid
Logic
wr_rd_counter
count_ctrl

FIG. 210

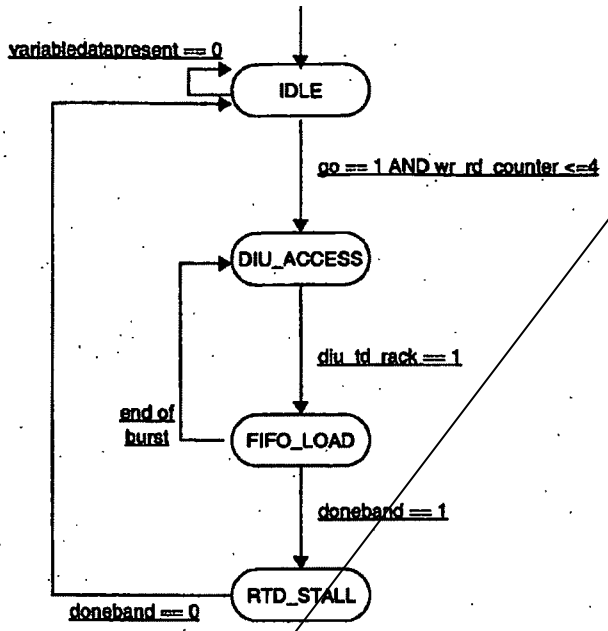


FIG. 211

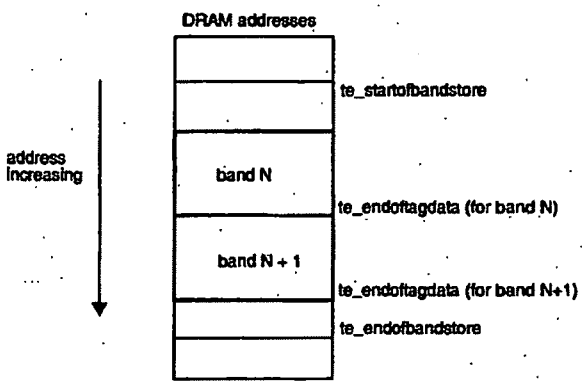


FIG. 212

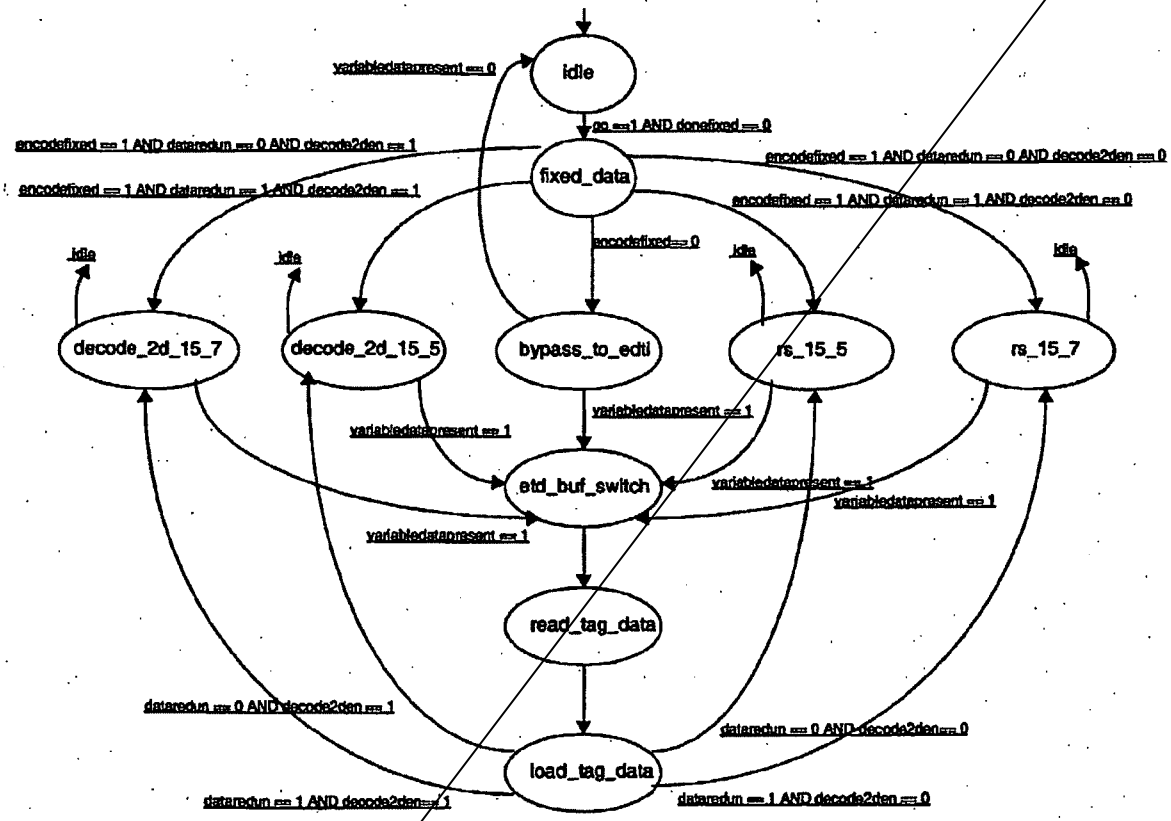

idle
variabledatapresent == 0
go == 1 AND donefixed == 0
fixed_data
encodefixed == 1 AND dataredun == 0 AND decode2den == 1
encodefixed == 1 AND dataredun == 0 AND decode2den == 0
encodefixed == 1 AND dataredun == 1 AND decode2den == 1
encodefixed == 1 AND dataredun == 1 AND decode2den == 0
encodefixed == 0
idle
idle
idle
idle
decode_2d_15_7
decode_2d_15_5
bypass_to_edti
rs_15_5
rs_15_7
variabledatapresent == 1
variabledatapresent == 1
variabledatapresent == 1
variabledatapresent == 1
variabledatapresent == 1
edti_buf_switch
read_tag_data
dataredun == 0 AND decode2den == 1
dataredun == 0 AND decode2den == 0
load_tag_data
dataredun == 1 AND decode2den == 1
dataredun == 1 AND decode2den == 0

FIG. 213

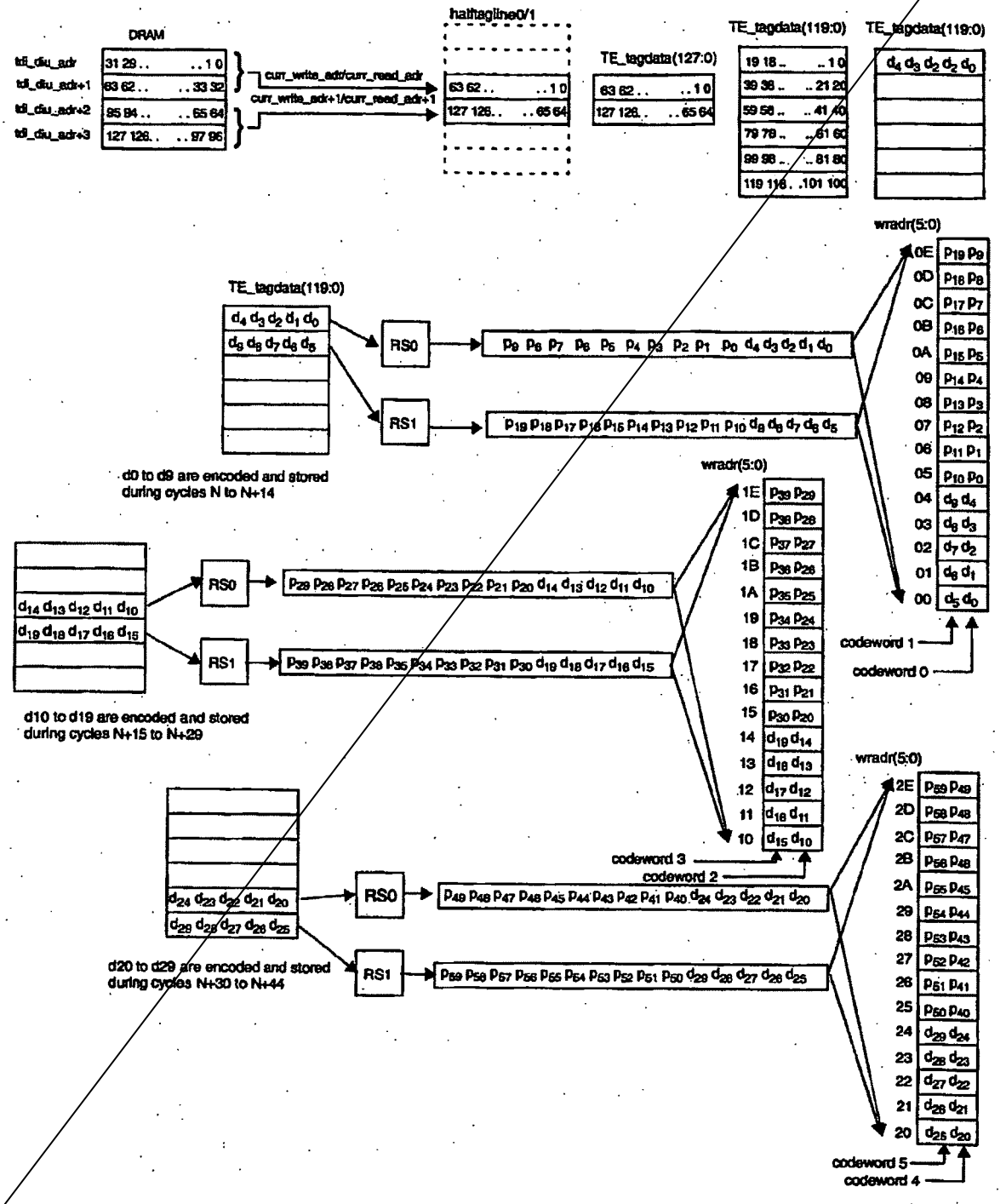

DRAM
tdl_diu_adr
tdl_diu_adr+1
tdl_diu_adr+2
tdl_diu_adr+3
curr_write_adr/curr_read_adr
curr_write_adr+1/curr_read_adr+1
halftagline0/1
TE_tagdata(127:0)
TE_tagdata(119:0)
TE_tagdata(119:0)
wradr(5:0)
RS0
RS1
d0 to d9 are encoded and stored during cycles N to N+14
d10 to d19 are encoded and stored during cycles N+15 to N+29
d20 to d29 are encoded and stored during cycles N+30 to N+44
codeword 0
codeword 1
codeword 2
codeword 3
codeword 4
codeword 5

FIG. 214

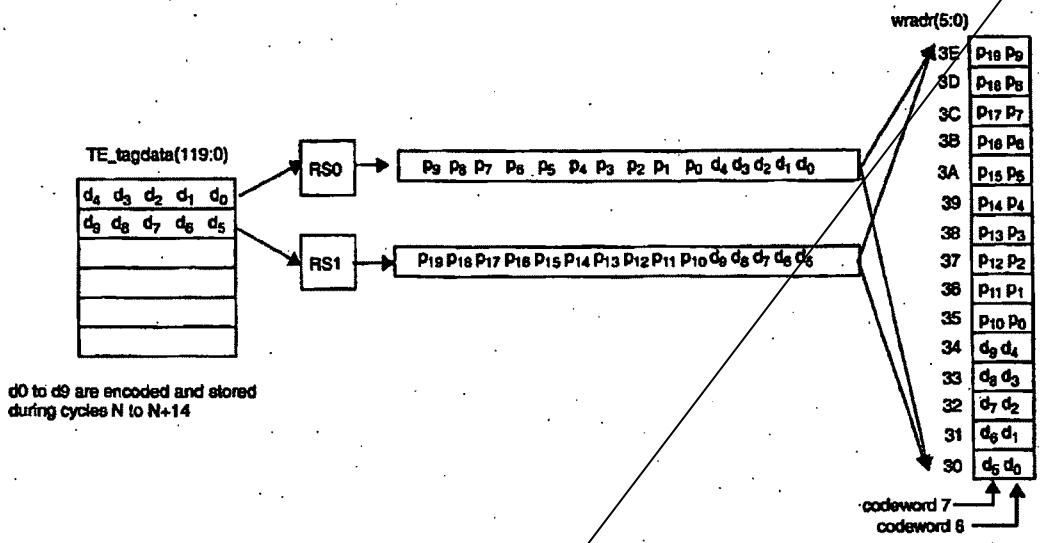


FIG. 215

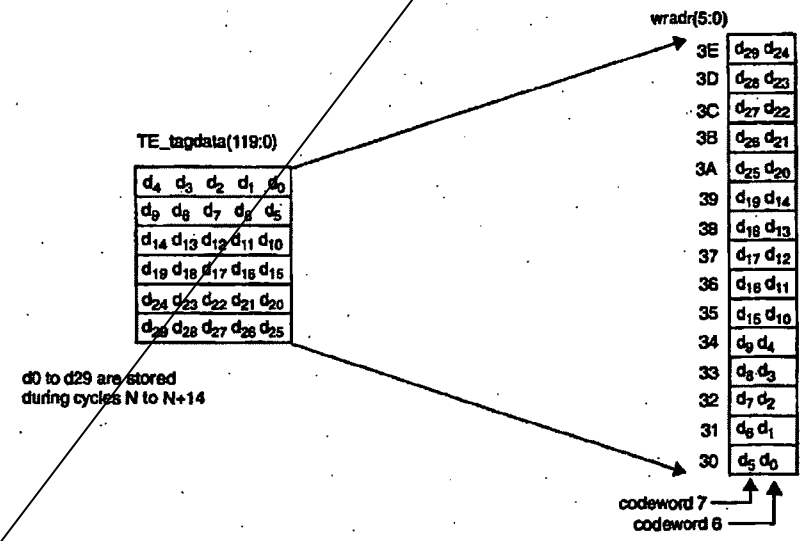


FIG. 216

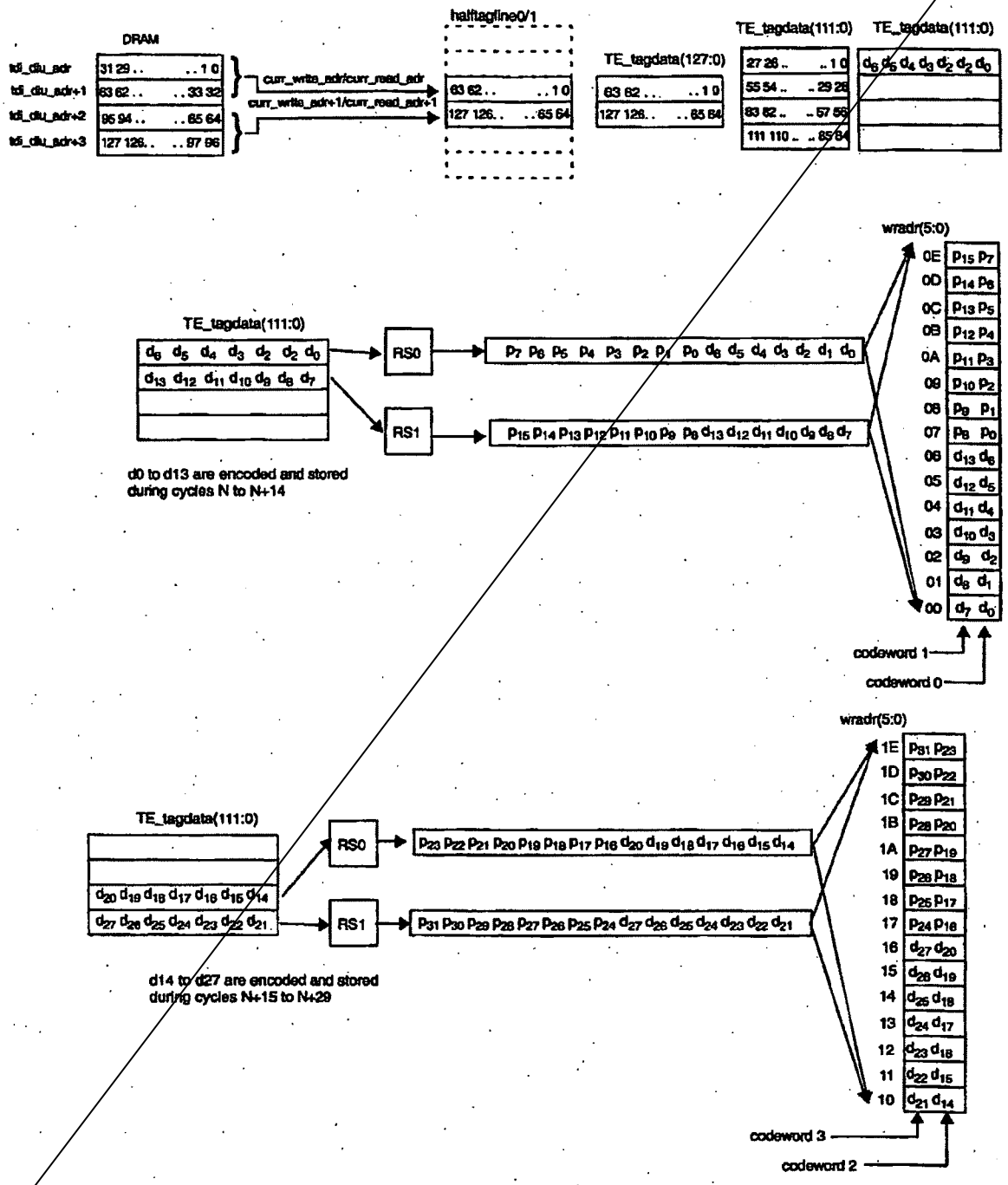

DRAM
tdi_dlu_adr
tdi_dlu_adr+1
tdi_dlu_adr+2
tdi_dlu_adr+3
curr_write_adr/curr_read_adr
curr_write_adr+1/curr_read_adr+1
halftagline0/1
TE_tagdata(127:0)
TE_tagdata(111:0)
TE_tagdata(111:0)
TE_tagdata(111:0)
RS0
RS1
d0 to d13 are encoded and stored during cycles N to N+14
wradr(5:0)
codeword 1
codeword 0
TE_tagdata(111:0)
RS0
RS1
d14 to d27 are encoded and stored during cycles N+15 to N+29
wradr(5:0)
codeword 3
codeword 2

FIG. 217

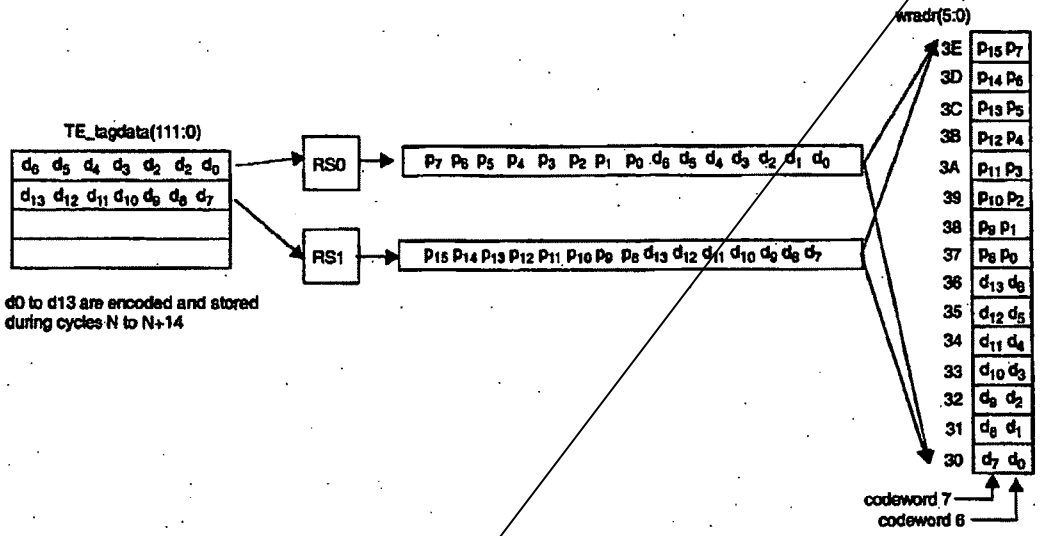

TE_tagdata(111:0)
d6 d5 d4 d3 d2 d2 d0
d13 d12 d11 d10 d9 d8 d7
RS0
RS1
p7 p6 p5 p4 p3 p2 p1 p0 d6 d5 d4 d3 d2 d1 d0
p15 p14 p13 p12 p11 p10 p9 p8 d13 d12 d11 d10 d9 d8 d7
wradr(5:0)
3E p15 p7
3D p14 p6
3C p13 p5
3B p12 p4
3A p11 p3
39 p10 p2
38 p9 p1
37 p8 p0
36 d13 d8
35 d12 d5
34 d11 d4
33 d10 d3
32 d9 d2
31 d8 d1
30 d7 d0
codeword 7
codeword 6
d0 to d13 are encoded and stored during cycles N to N+14

FIG. 218

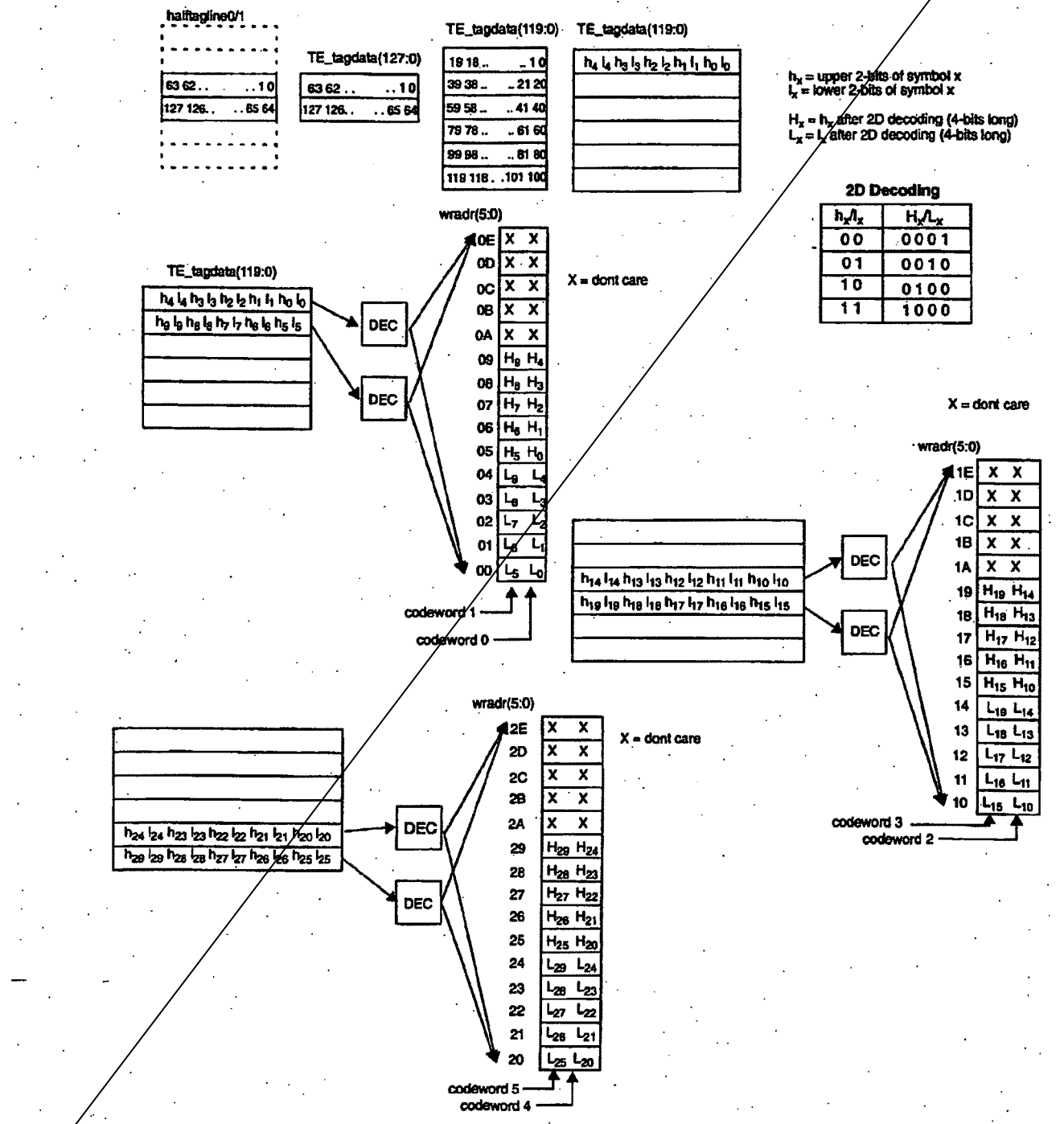

halftagline0/1
63 62 . . . . 1 0
127 126. . . . 65 64
TE_tagdata(127:0)
63 62 . . . . 1 0
127 126. . . . 65 64
TE_tagdata(119:0)
19 18 . . . . 1 0
39 38 . . . . 21 20
59 58 . . . . 41 40
79 78 . . . . 61 60
99 98 . . . . 81 80
119 118 . . . 101 100
TE_tagdata(119:0)
h4 l4 h3 l3 h2 l2 h1 l1 h0 l0
hx = upper 2-bits of symbol x
lx = lower 2-bits of symbol x
Hx = hx after 2D decoding (4-bits long)
Lx = lx after 2D decoding (4-bits long)
2D Decoding
hx/lx
Hx/Lx
00
0001
01
0010
10
0100
11
1000
TE_tagdata(119:0)
h4 l4 h3 l3 h2 l2 h1 l1 h0 l0
h9 l9 h8 l8 h7 l7 h6 l6 h5 l5
DEC
DEC
wradr(5:0)
0E X X
0D X X
0C X X
0B X X
0A X X
09 H9 H4
08 H8 H3
07 H7 H2
06 H6 H1
05 H5 H0
04 L9 L4
03 L8 L3
02 L7 L2
01 L6 L1
00 L5 L0
X = dont care
codeword 1
codeword 0
h14 l14 h13 l13 h12 l12 h11 l11 h10 l10
h19 l19 h18 l18 h17 l17 h16 l16 h15 l15
DEC
DEC
X = dont care
wradr(5:0)
1E X X
1D X X
1C X X
1B X X
1A X X
19 H19 H14
18 H18 H13
17 H17 H12
16 H16 H11
15 H15 H10
14 L19 L14
13 L18 L13
12 L17 L12
11 L16 L11
10 L15 L10
codeword 3
codeword 2
h24 l24 h23 l23 h22 l22 h21 l21 h20 l20
h29 l29 h28 l28 h27 l27 h26 l26 h25 l25
DEC
DEC
wradr(5:0)
2E X X
2D X X
2C X X
2B X X
2A X X
29 H29 H24
28 H28 H23
27 H27 H22
26 H26 H21
25 H25 H20
24 L29 L24
23 L28 L23
22 L27 L22
21 L26 L21
20 L25 L20
X = dont care
codeword 5
codeword 4

FIG. 219

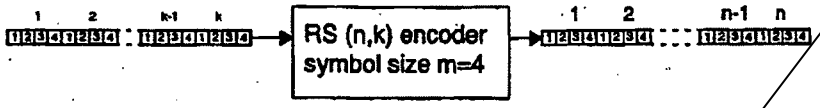


FIG. 220

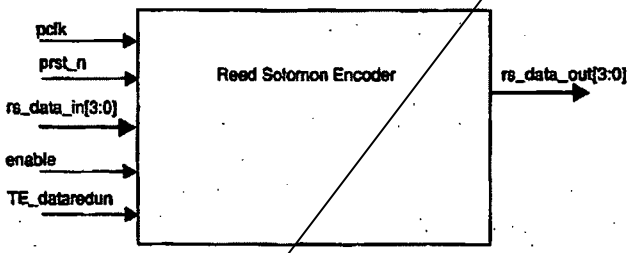


FIG. 221

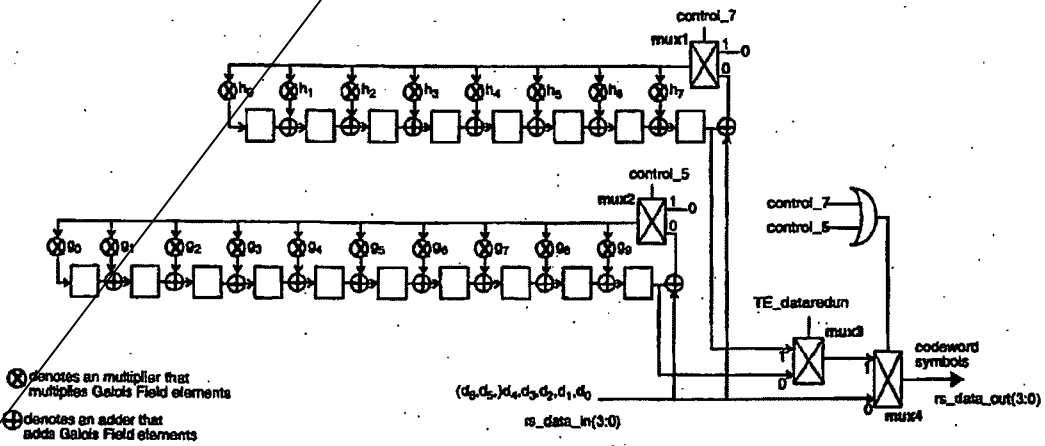


FIG. 222

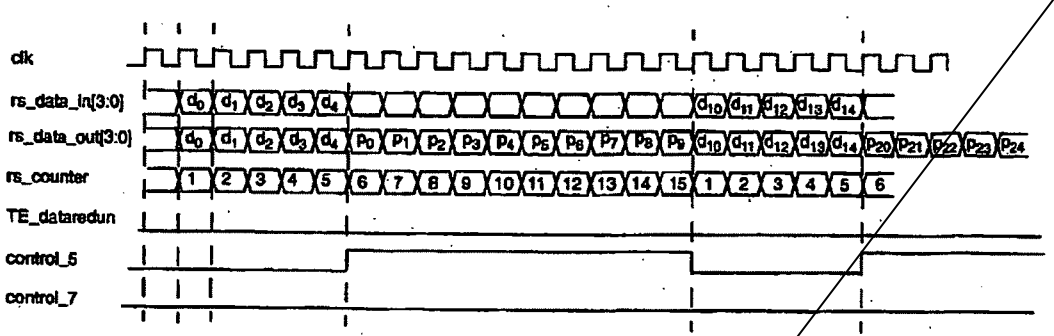


FIG. 223

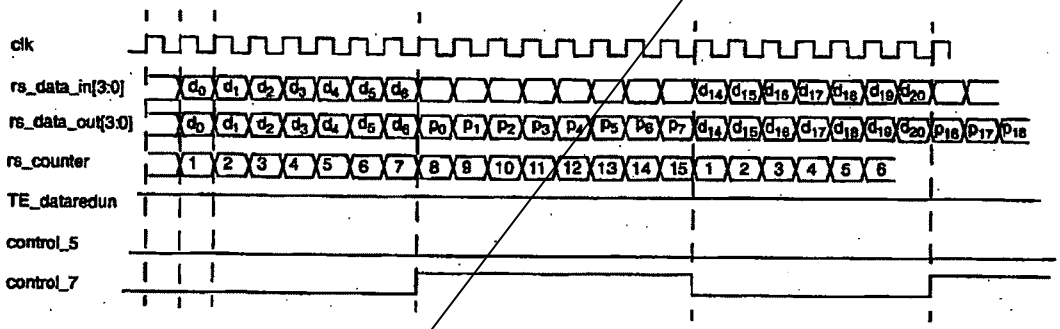


FIG. 224

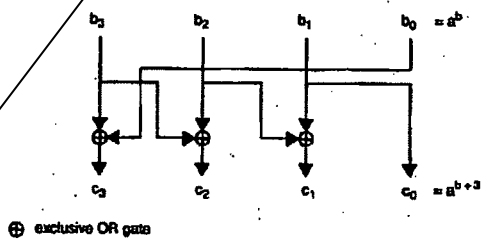


FIG. 225

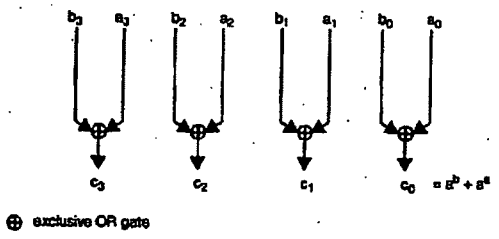


FIG. 226

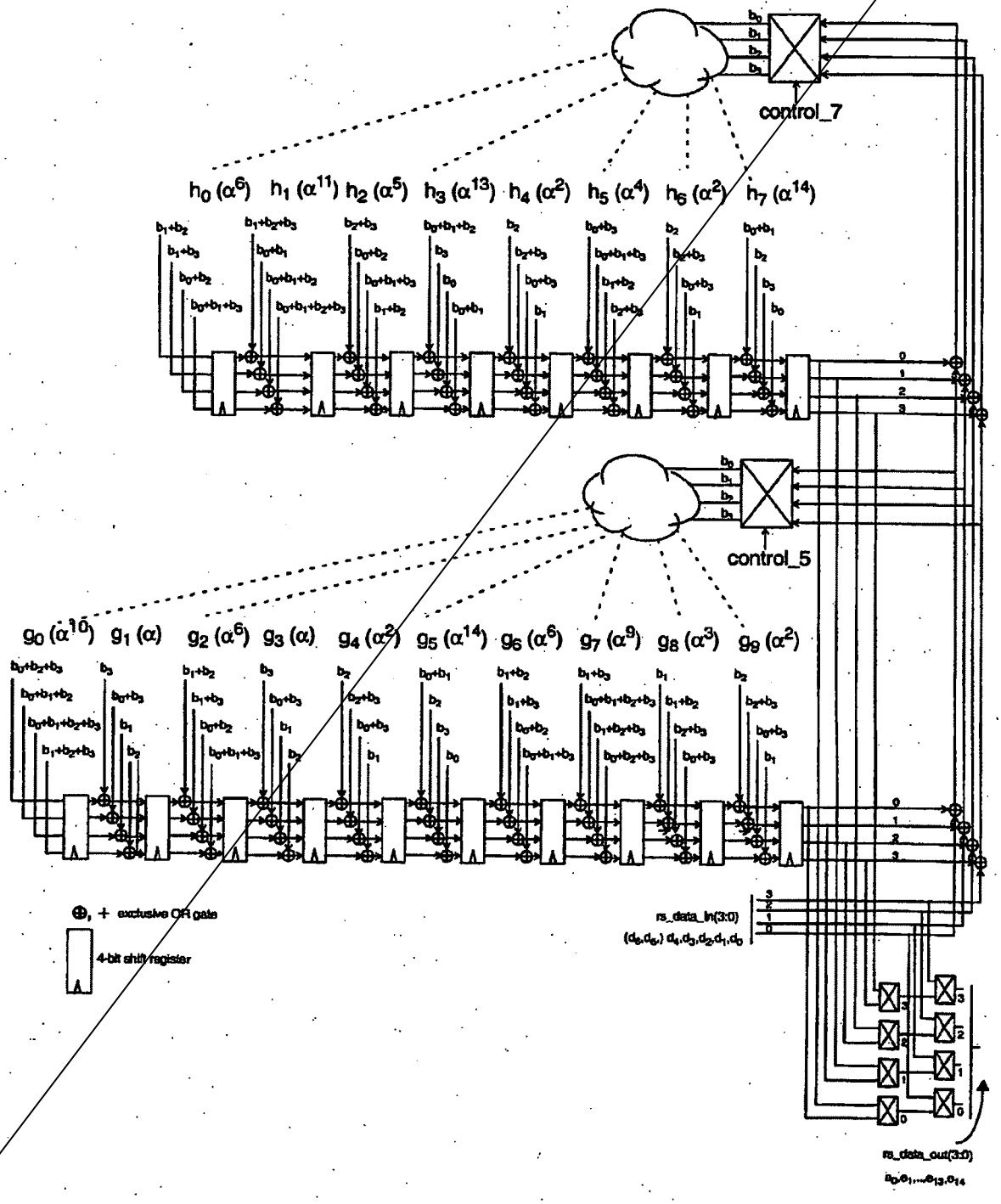

control_7
h0 (α6) h1 (α11) h2 (α5) h3 (α13) h4 (α2) h5 (α4) h6 (α2) h7 (α14)
control_5
g0 (α10) g1 (α) g2 (α6) g3 (α) g4 (α2) g5 (α14) g6 (α6) g7 (α9) g8 (α3) g9 (α2)
⊕, + exclusive OR gate
4-bit shift register
rs_data_in(3:0)
(d5,d6,) d4,d3,d2,d1,d0
rs_data_out(3:0)
e0,e1,...e13,e14

FIG. 227

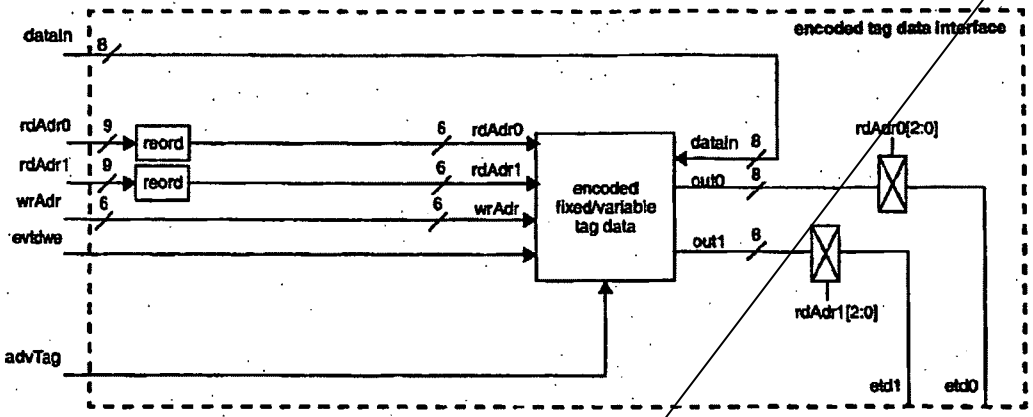


FIG. 228

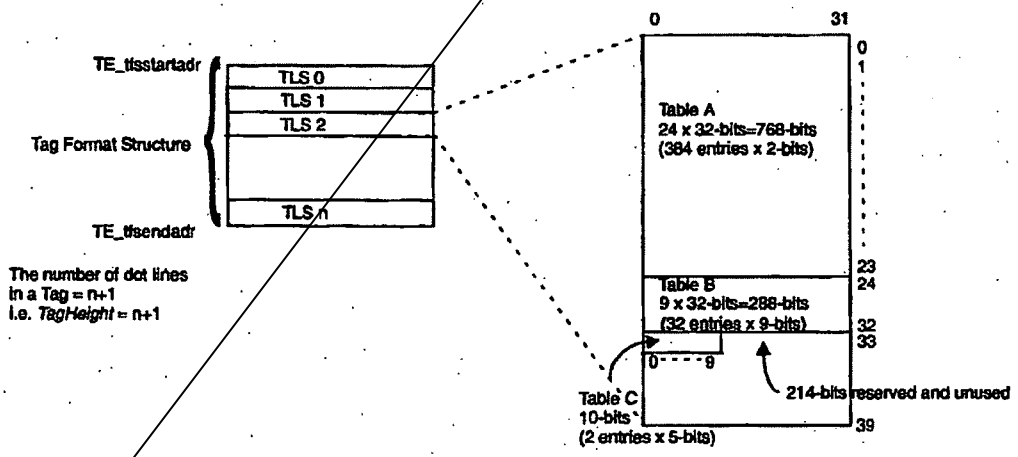


FIG. 229

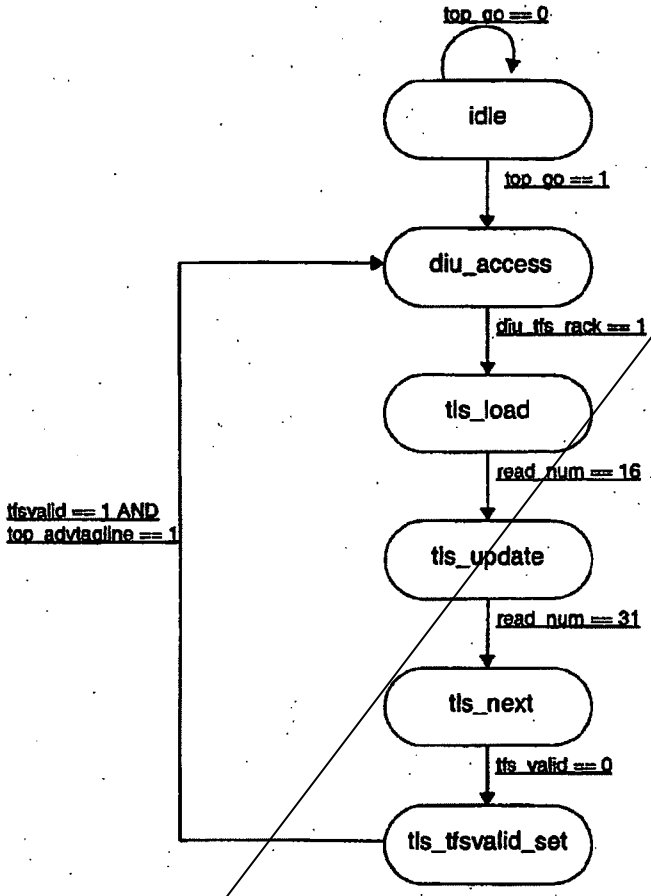

top_go == 0
idle
top_go == 1
diu_access
diu_tfs_rack == 1
tls_load
read_num == 16
tls_update
read_num == 31
tls_next
tfs_valid == 0
tls_tfsvalid_set
tfsvalid == 1 AND
top_advtagline == 1

FIG. 230

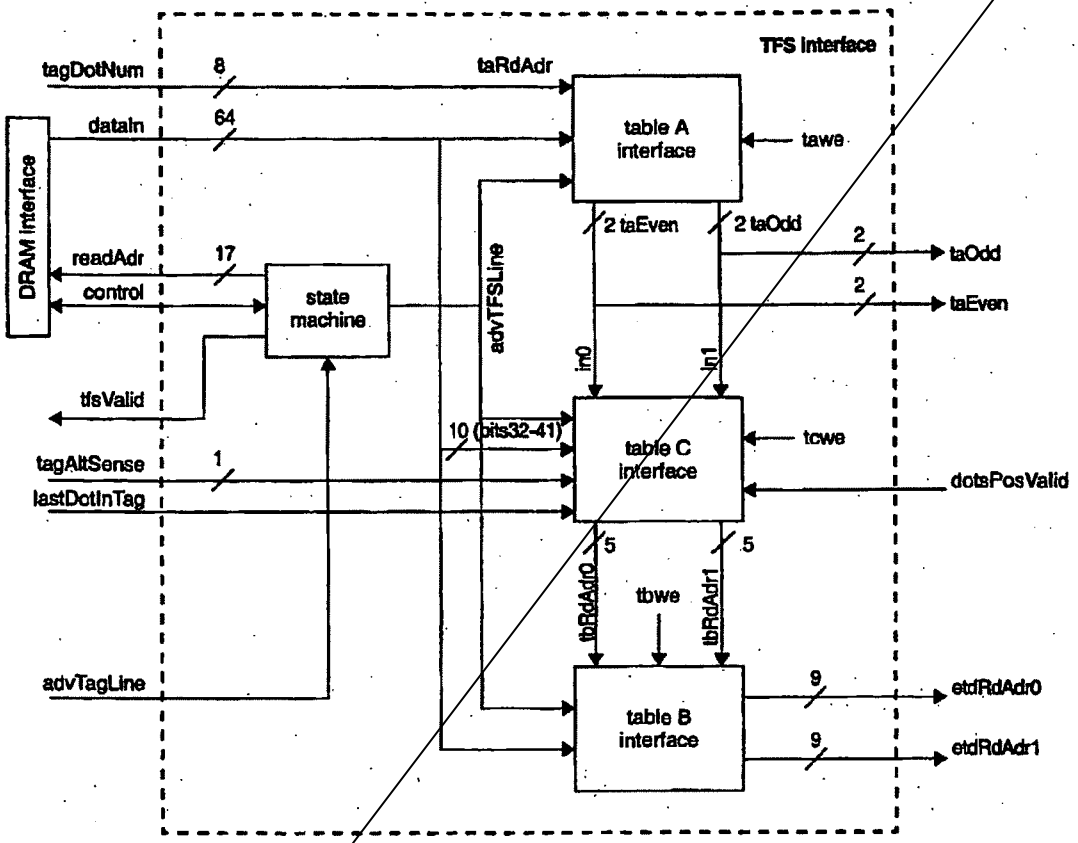


FIG. 231

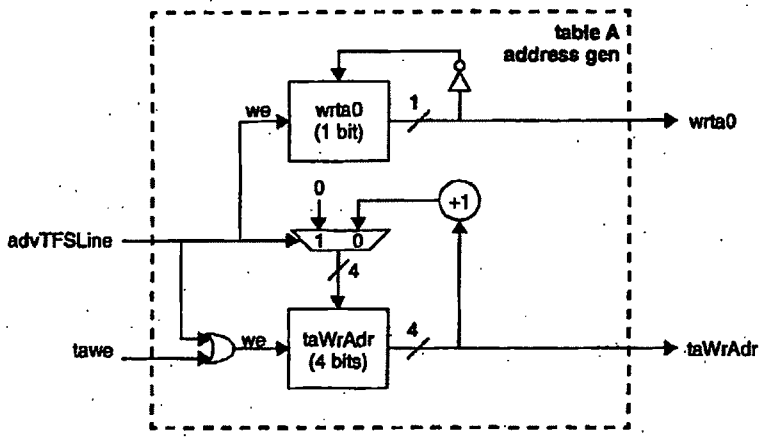


FIG. 232

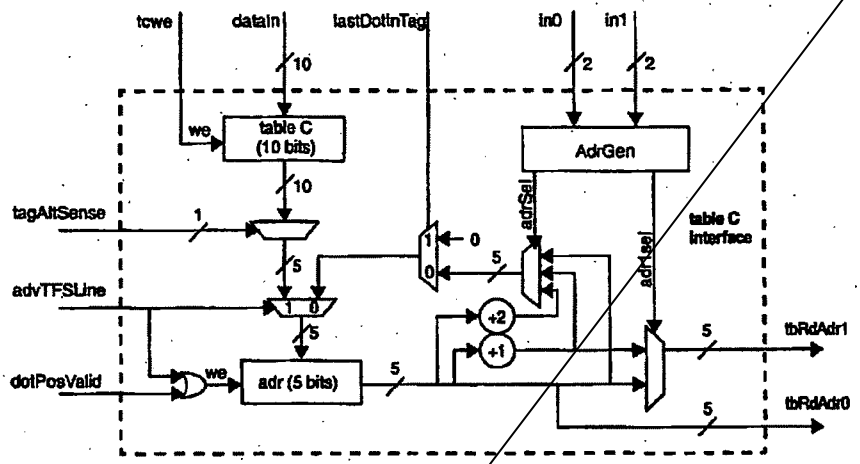


FIG. 233

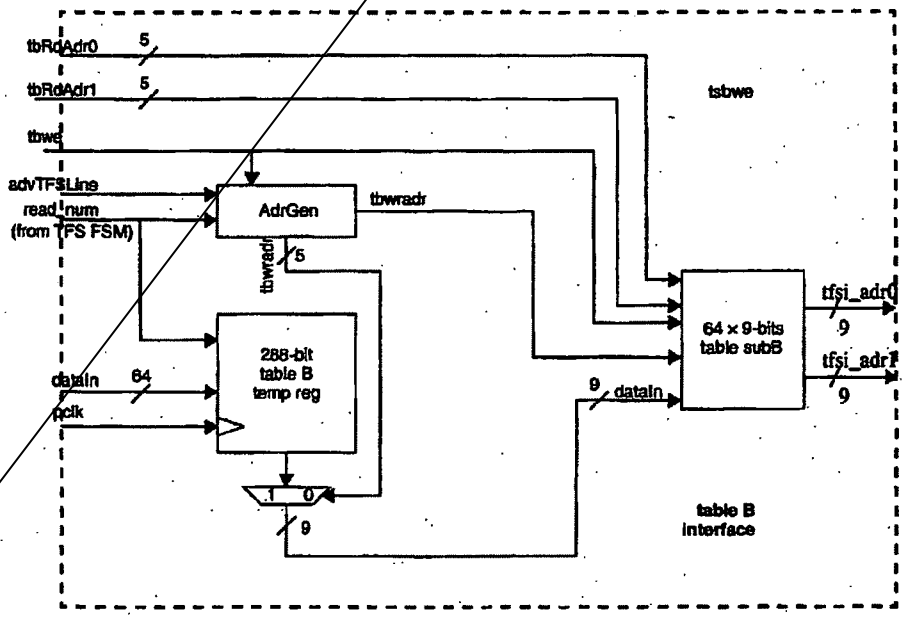


FIG. 234

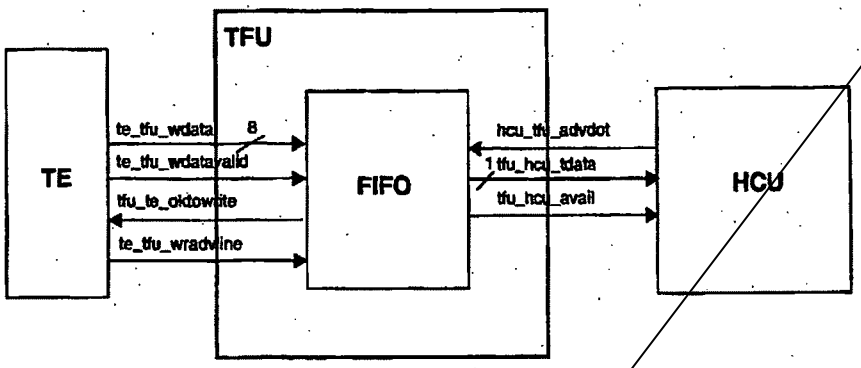


FIG. 235

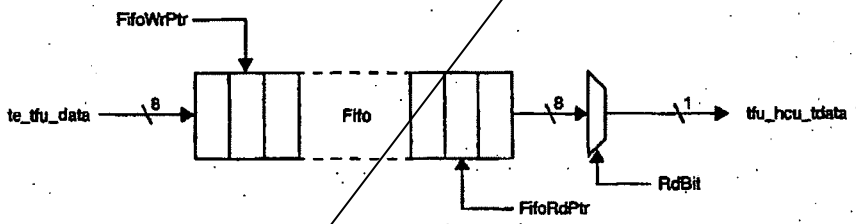


FIG. 236

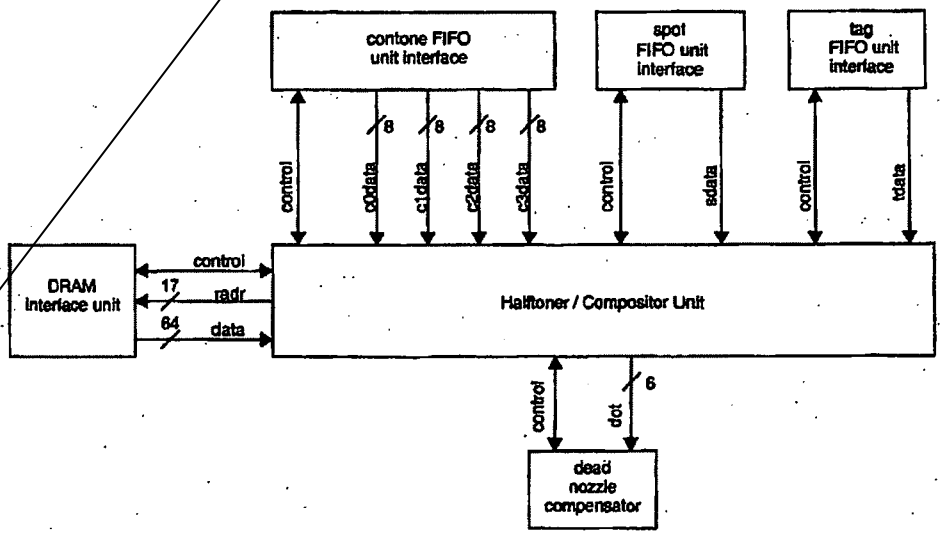


FIG. 237

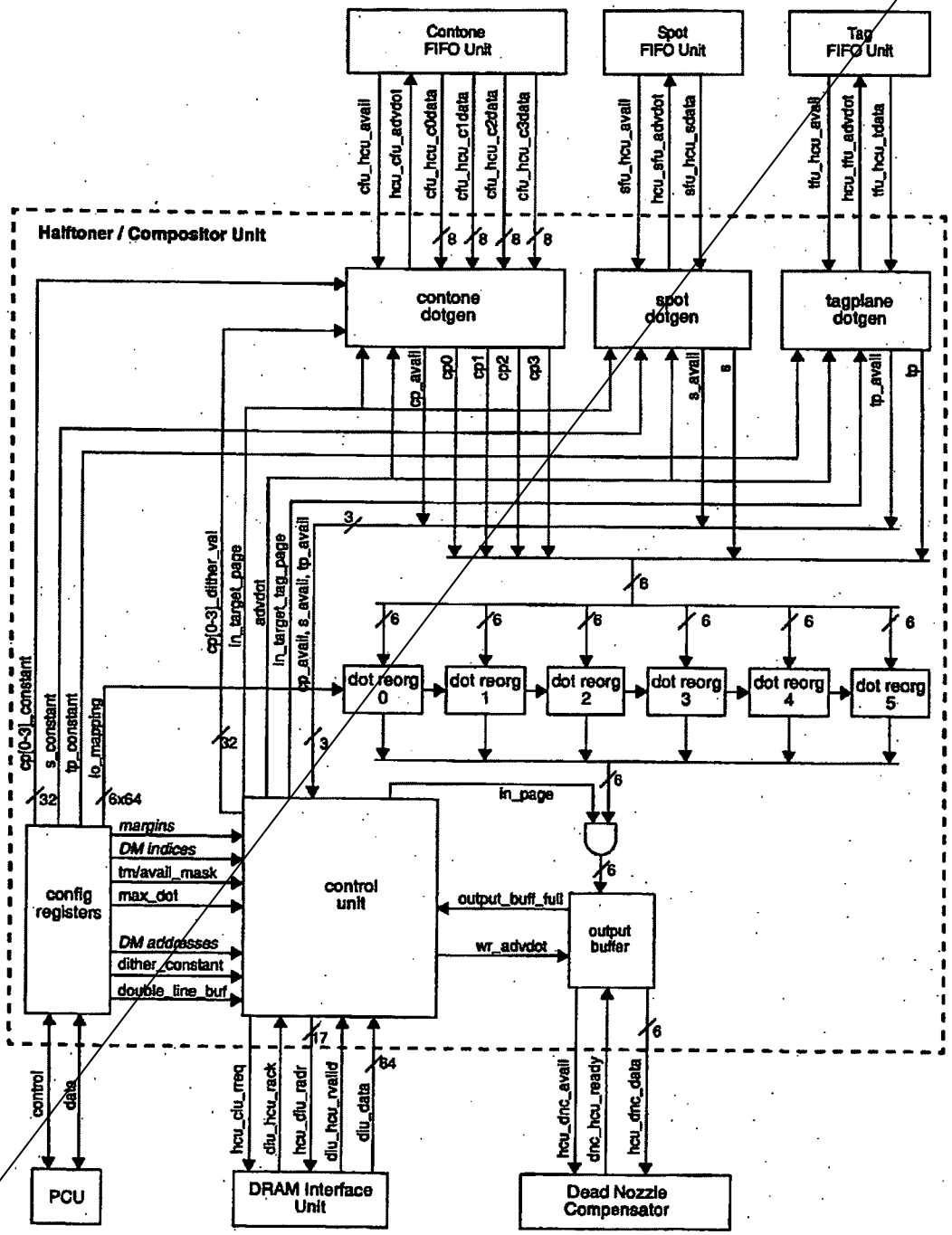

Contone FIFO Unit
Spot FIFO Unit
Tag FIFO Unit
cfu_hcu_avail
hcu_cfu_advdot
cfu_hcu_c0data
cfu_hcu_c1data
cfu_hcu_c2data
cfu_hcu_c3data
sfu_hcu_avail
hcu_sfu_advdot
sfu_hcu_sdata
tfu_hcu_avail
hcu_tfu_advdot
tfu_hcu_tdata
Halftoner / Compositor Unit
contone dotgen
spot dotgen
tagplane dotgen
cp_avail
cp0
cp1
cp2
cp3
s_avail
s
tp_avail
tp
cp[0-3]_constant
s_constant
tp_constant
lo_mapping
cp[0-3]_dither_val
in_target_page
advdot
in_target_tag_page
cp_avail, s_avail, tp_avail
dot reorg 0
dot reorg 1
dot reorg 2
dot reorg 3
dot reorg 4
dot reorg 5
in_page
config registers
margins
DM indices
tm/avail_mask
max_dot
DM addresses
dither_constant
double_line_buf
control unit
output_buff_full
wr_advdot
output buffer
control
data
hcu_diu_req
diu_hcu_rack
hcu_diu_radr
diu_hcu_rvalid
diu_data
hcu_dnc_avail
dnc_hcu_ready
hcu_dnc_data
PCU
DRAM Interface Unit
Dead Nozzle Compensator

FIG. 238

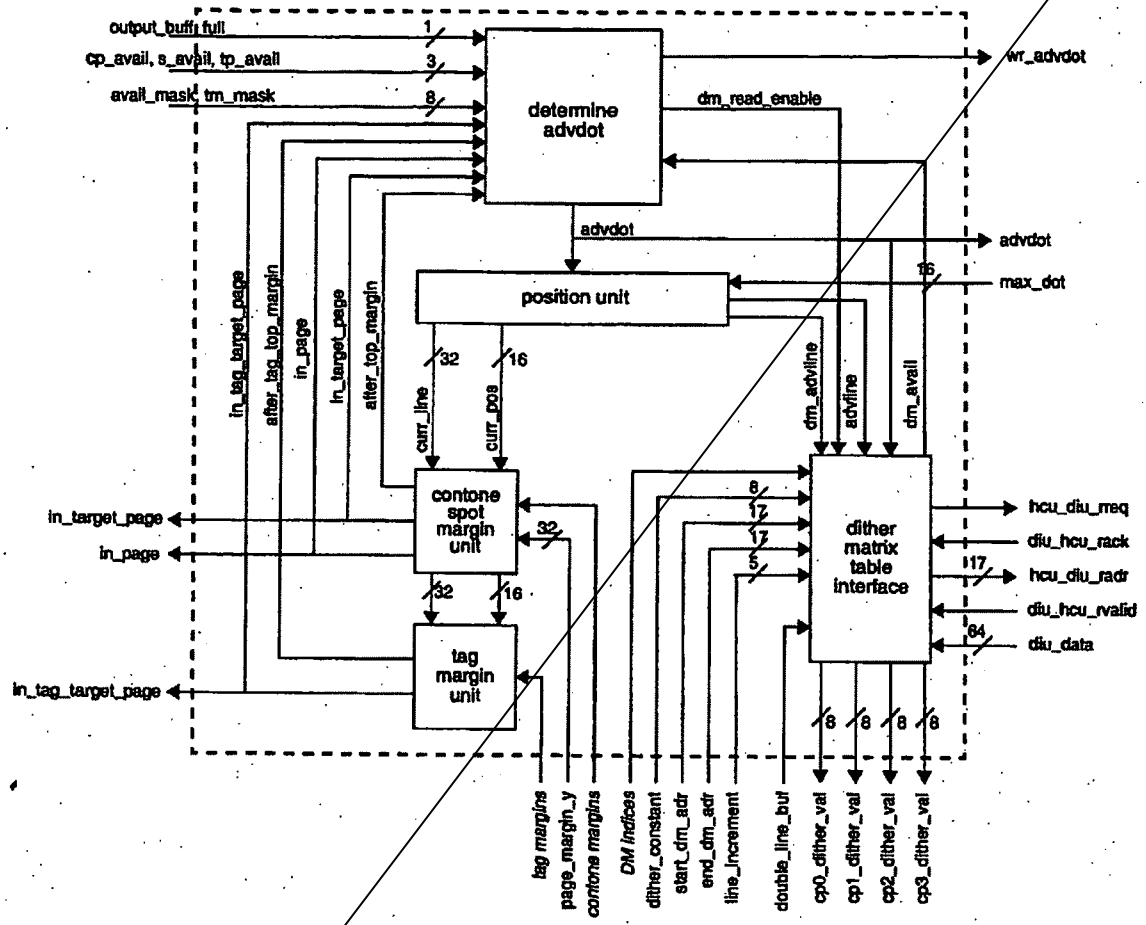

output_buff_full
cp_avail, s_avail, tp_avail
avail_mask, tm_mask
determine advdot
wr_advdot
dm_read_enable
advdot
advdot
position unit
max_dot
in_tag_target_page
after_tag_top_margin
in_page
in_target_page
after_top_margin
curr_line
curr_pos
dm_advline
advline
dm_avail
contone spot margin unit
tag margin unit
dither matrix table interface
in_target_page
in_page
in_tag_target_page
hcu_diu_req
diu_hcu_rack
hcu_diu_radr
diu_hcu_rvalid
diu_data
tag margins
page_margin_y
contone margins
DM indices
dither_constant
start_dm_adr
end_dm_adr
line_increment
double_line_buf
cp0_dither_val
cp1_dither_val
cp2_dither_val
cp3_dither_val

FIG. 239

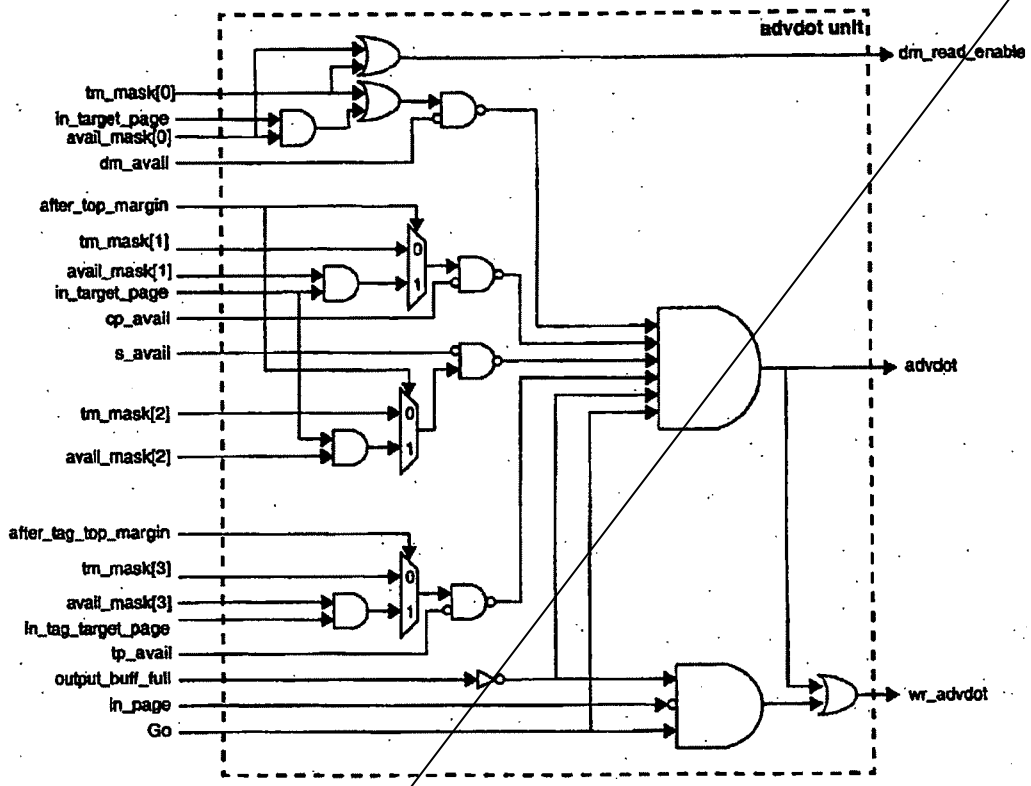


FIG. 240

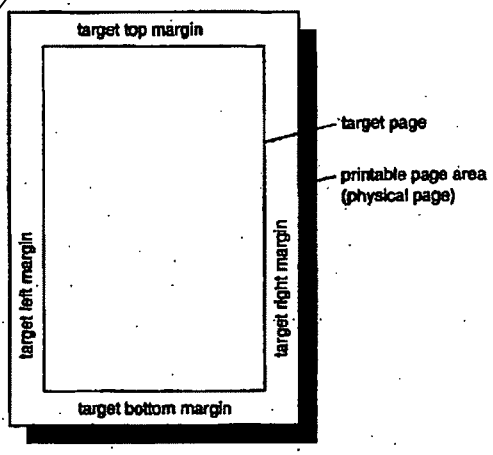


FIG. 241

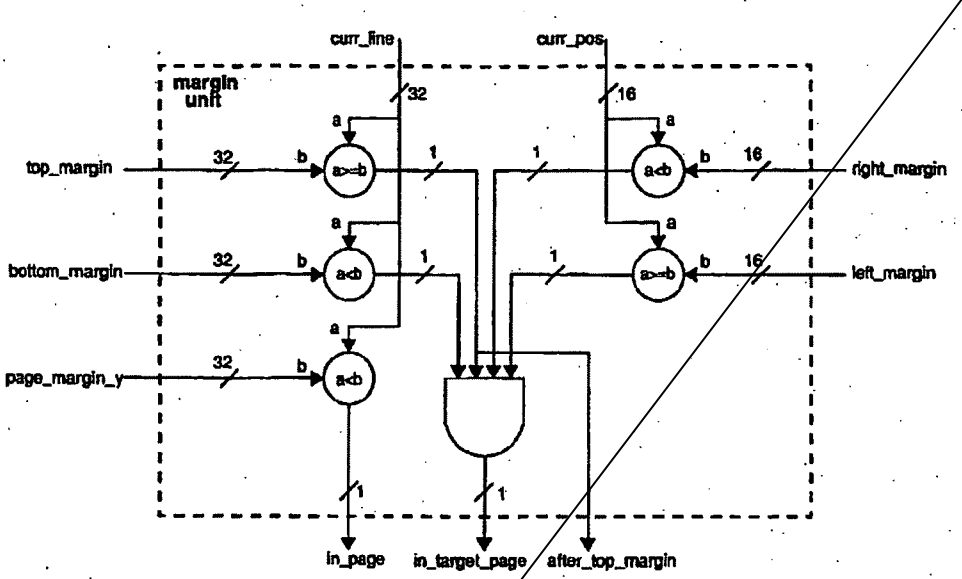


FIG. 242

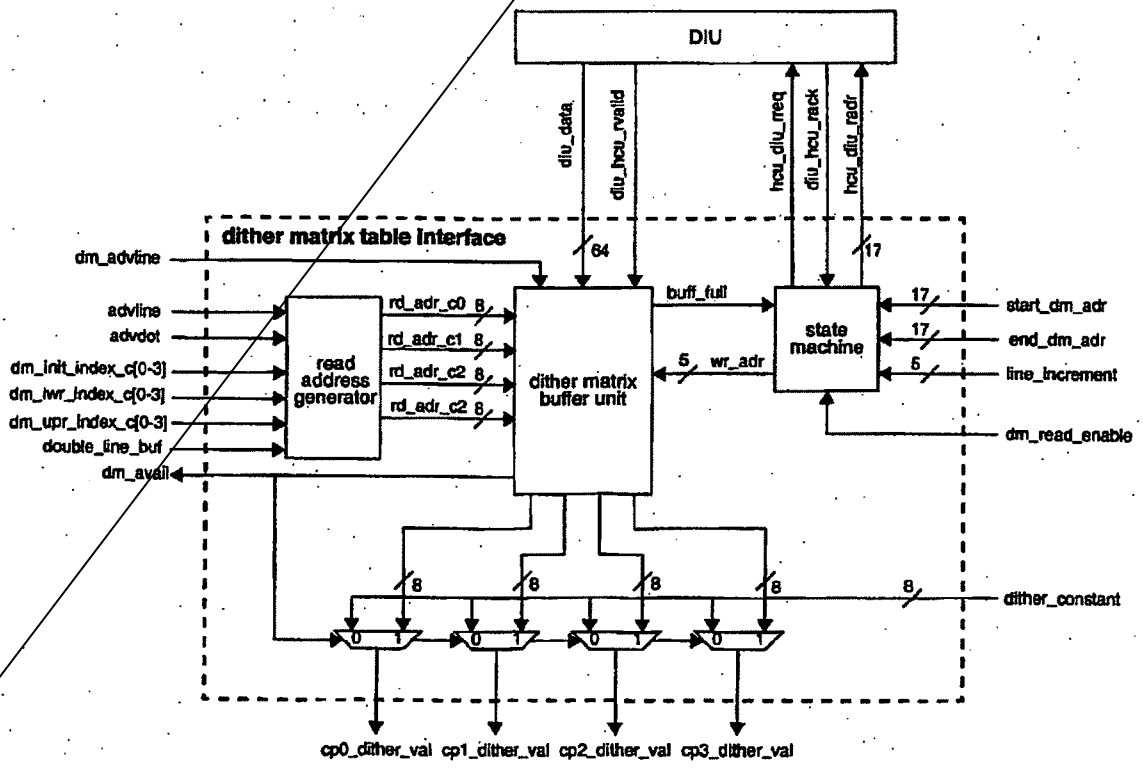


FIG. 243

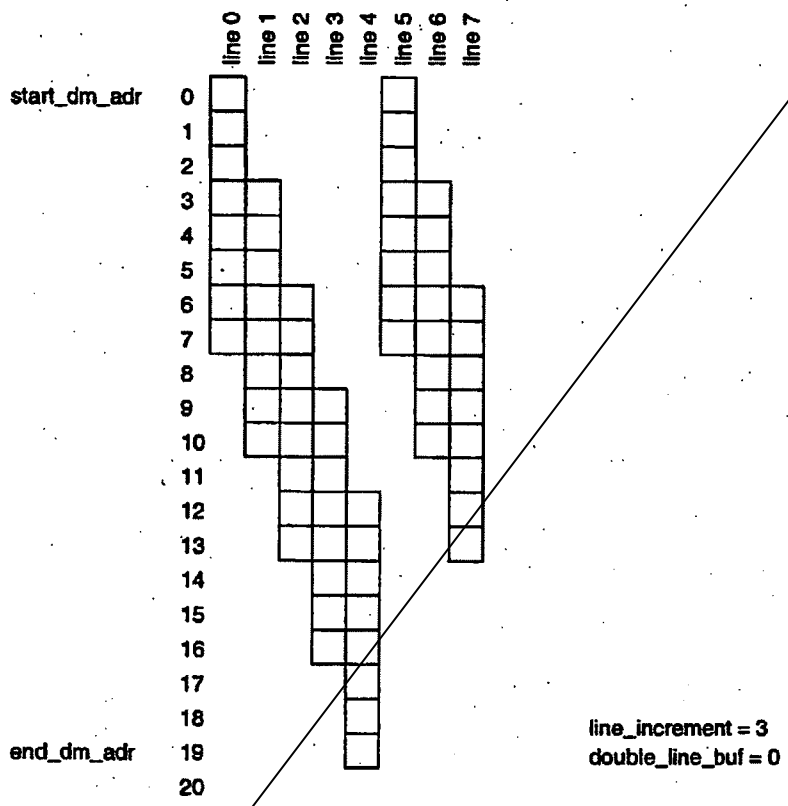

line 0
line 1
line 2
line 3
line 4
line 5
line 6
line 7
start_dm_adr
end_dm_adr
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
line_increment = 3
double_line_buf = 0

FIG. 244

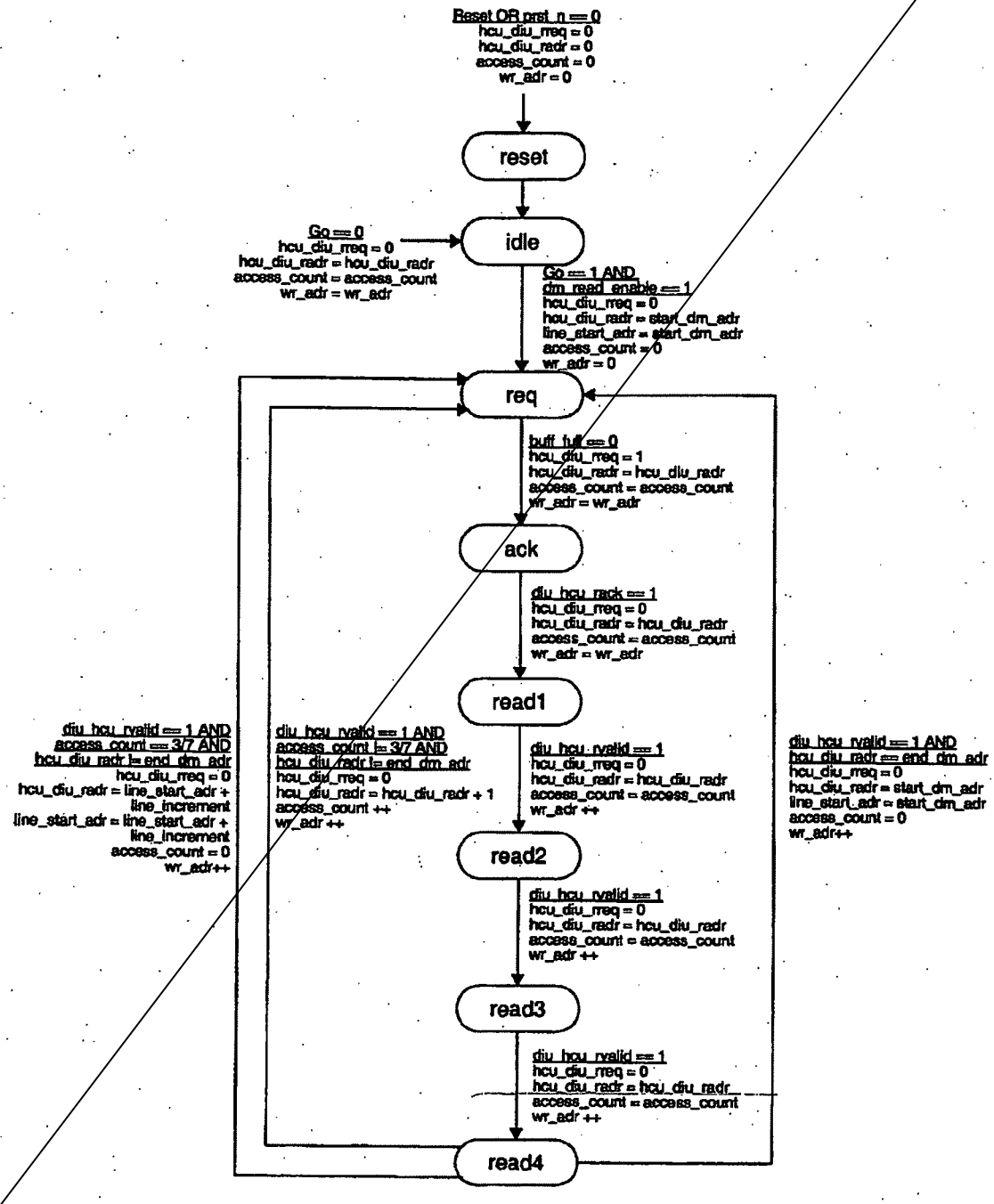

Reset OR prst_n == 0
hcu_diu_rreq = 0
hcu_diu_radr = 0
access_count = 0
wr_adr = 0
reset
idle
Go == 0
hcu_diu_rreq = 0
hcu_diu_radr = hcu_diu_radr
access_count = access_count
wr_adr = wr_adr
Go == 1 AND
dm_read_enable == 1
hcu_diu_rreq = 0
hcu_diu_radr = start_dm_adr
line_start_adr = start_dm_adr
access_count = 0
wr_adr = 0
req
buff_full == 0
hcu_diu_rreq = 1
hcu_diu_radr = hcu_diu_radr
access_count = access_count
wr_adr = wr_adr
ack
diu_hcu_rack == 1
hcu_diu_rreq = 0
hcu_diu_radr = hcu_diu_radr
access_count = access_count
wr_adr = wr_adr
read1
diu_hcu_rvalid == 1 AND
access_count == 3/7 AND
hcu_diu_radr != end_dm_adr
hcu_diu_rreq = 0
hcu_diu_radr = line_start_adr +
line_increment
line_start_adr = line_start_adr +
line_increment
access_count = 0
wr_adr++
diu_hcu_rvalid == 1 AND
access_count != 3/7 AND
hcu_diu_radr != end_dm_adr
hcu_diu_rreq = 0
hcu_diu_radr = hcu_diu_radr + 1
access_count ++
wr_adr ++
diu_hcu_rvalid == 1
hcu_diu_rreq = 0
hcu_diu_radr = hcu_diu_radr
access_count = access_count
wr_adr ++
diu_hcu_rvalid == 1 AND
hcu_diu_radr == end_dm_adr
hcu_diu_rreq = 0
hcu_diu_radr = start_dm_adr
line_start_adr = start_dm_adr
access_count = 0
wr_adr++
read2
diu_hcu_rvalid == 1
hcu_diu_rreq = 0
hcu_diu_radr = hcu_diu_radr
access_count = access_count
wr_adr ++
read3
diu_hcu_rvalid == 1
hcu_diu_rreq = 0
hcu_diu_radr = hcu_diu_radr
access_count = access_count
wr_adr ++
read4

FIG. 245

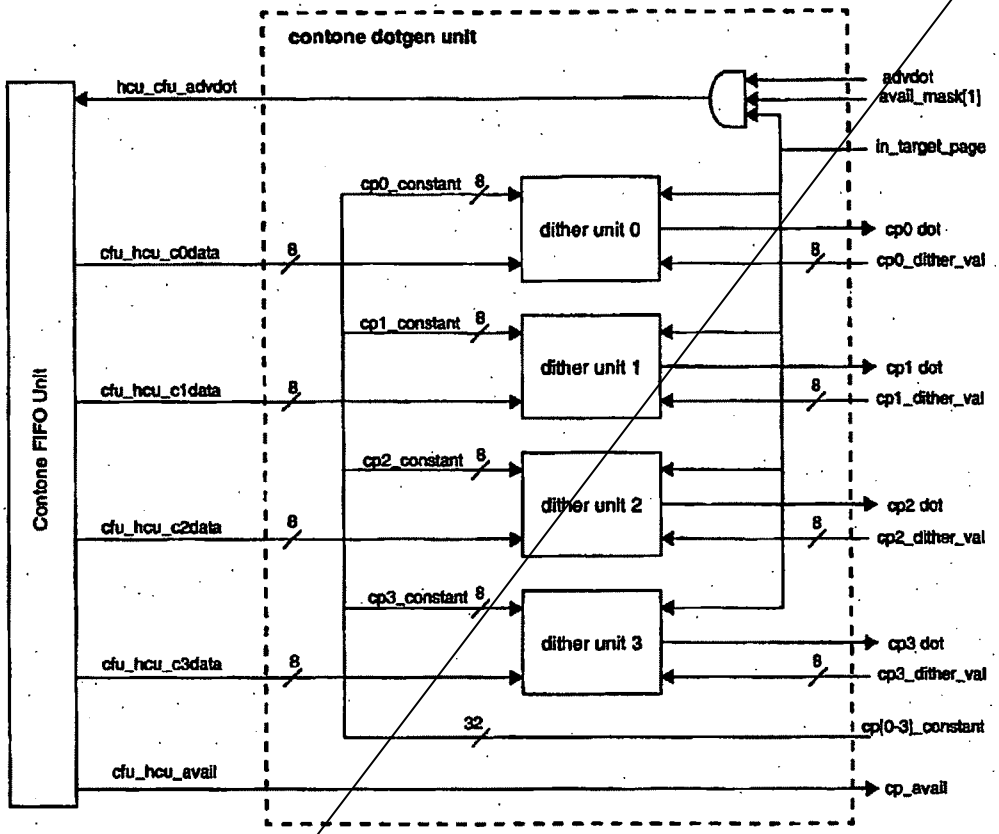


FIG. 246

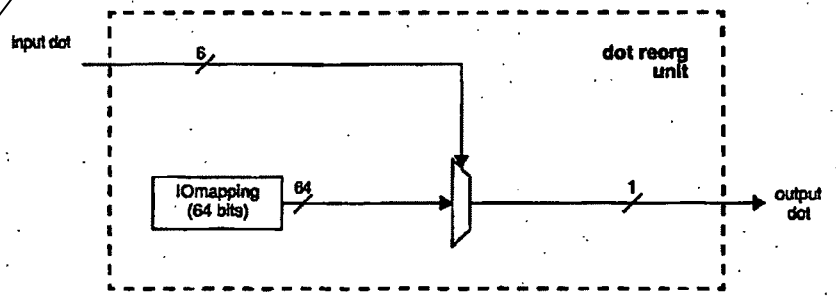


FIG. 247

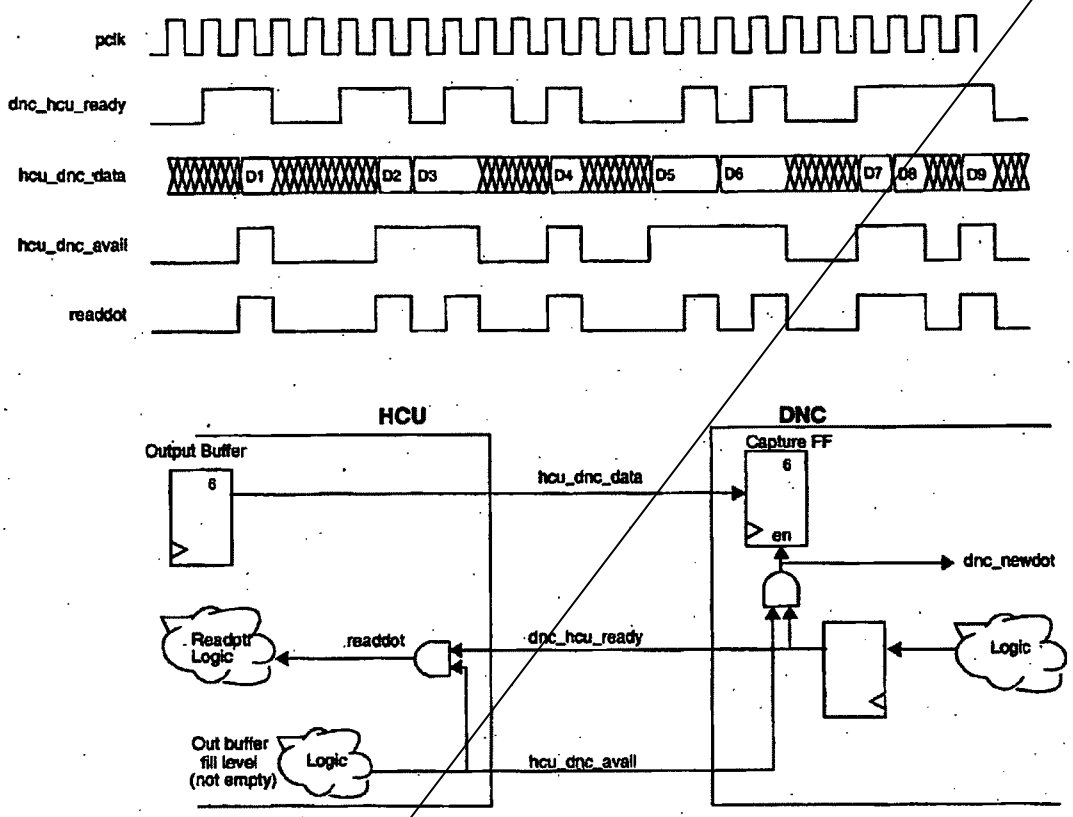


FIG. 248

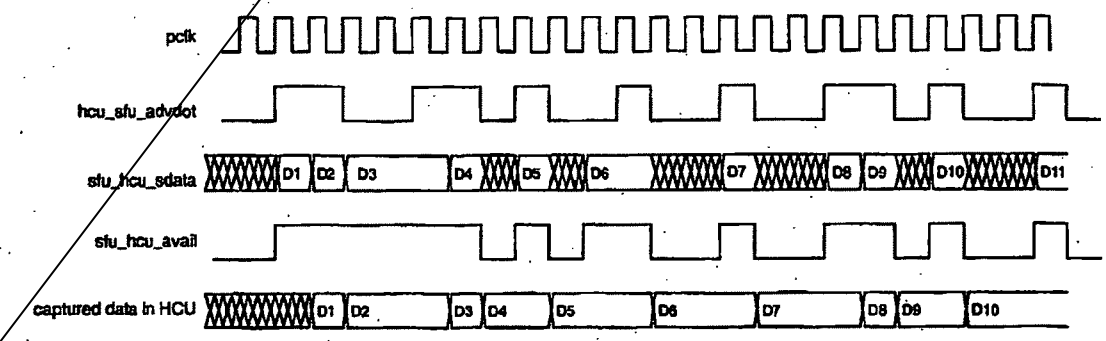


FIG. 249

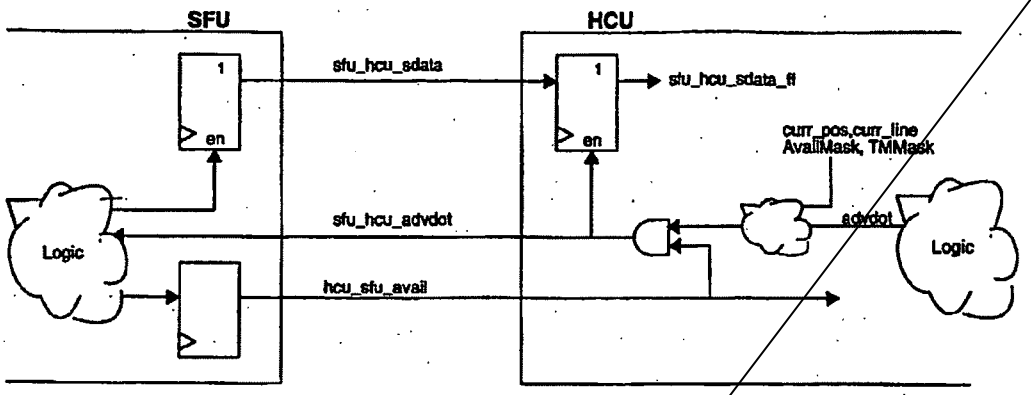


FIG. 250

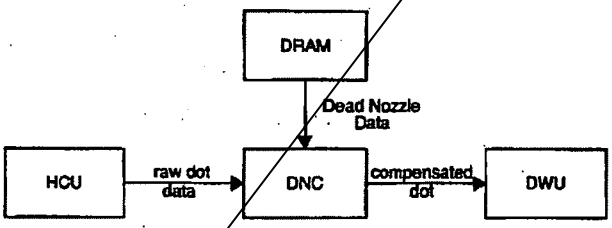


FIG. 251

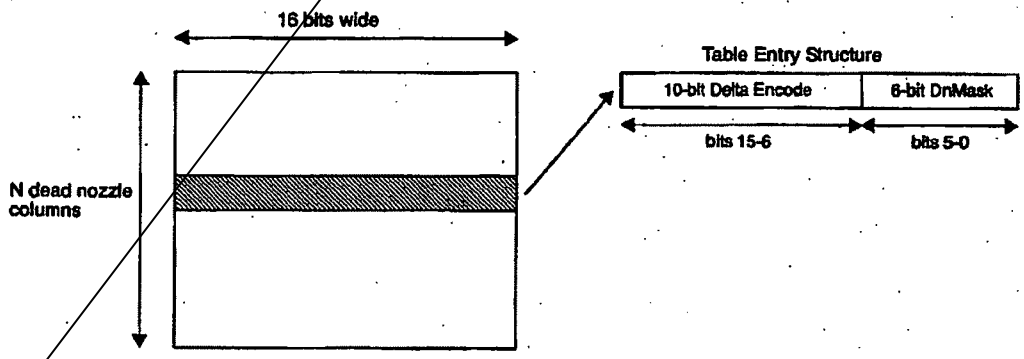


FIG. 252

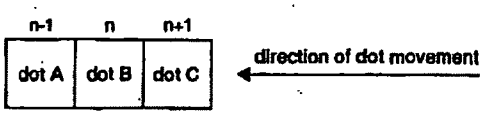


FIG. 253

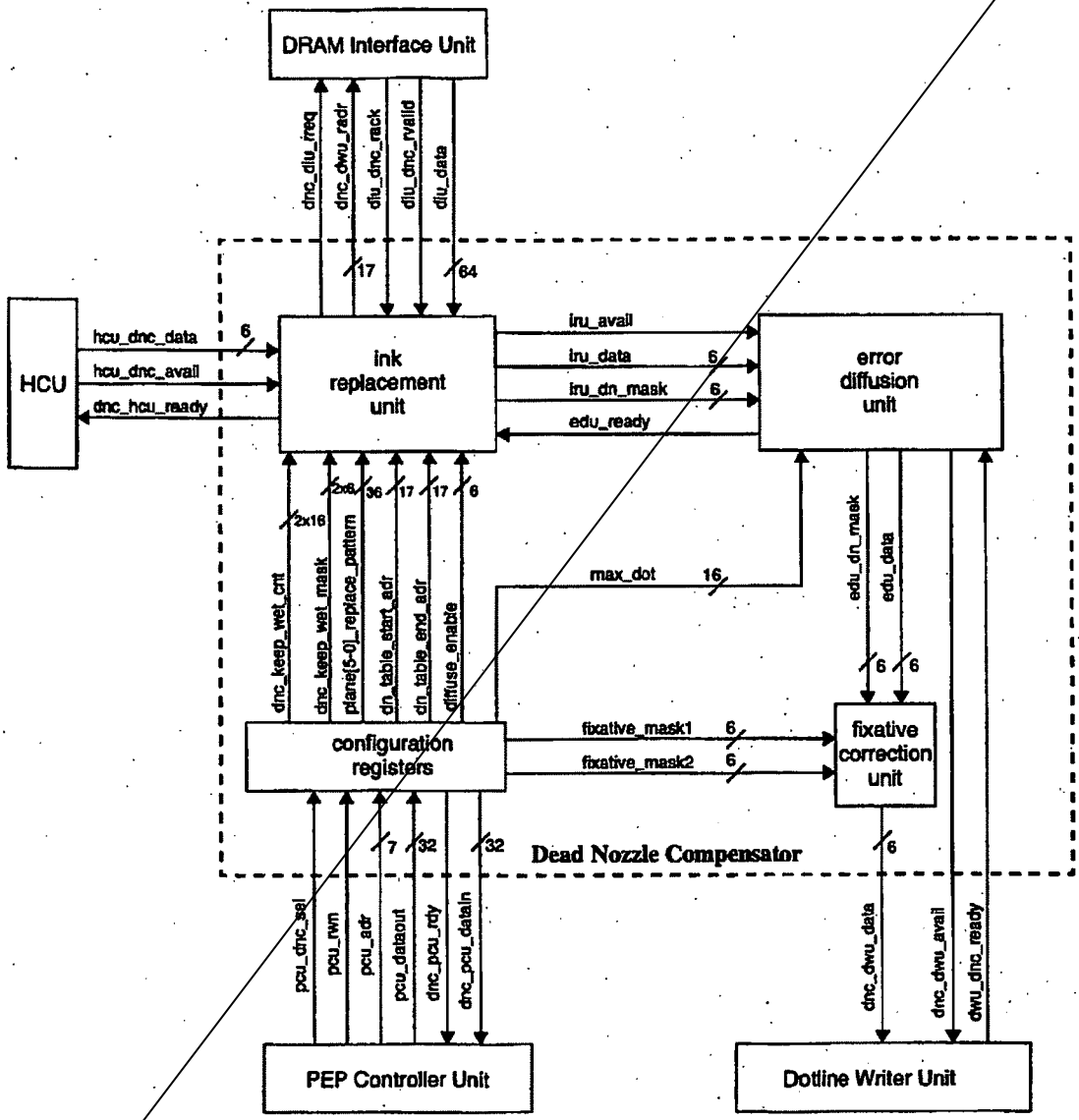

DRAM Interface Unit
dnc_diu_rreq
dnc_dwu_radr
diu_dnc_rack
diu_dnc_rvalid
diu_data
17
64
HCU
hcu_dnc_data
hcu_dnc_avail
dnc_hcu_ready
6
ink replacement unit
iru_avail
iru_data
iru_dn_mask
edu_ready
6
6
error diffusion unit
2x8
36
17
17
6
2x16
dnc_keep_wet_cnt
dnc_keep_wet_mask
plane[5-0]_replace_pattern
dn_table_start_adr
dn_table_end_adr
diffuse_enable
max_dot
16
edu_dn_mask
edu_data
6
6
configuration registers
fixative_mask1
fixative_mask2
6
6
fixative correction unit
7
32
32
Dead Nozzle Compensator
6
pcu_dnc_sel
pcu_rwn
pcu_adr
pcu_dataout
dnc_pcu_rdy
dnc_pcu_datain
dnc_dwu_data
dnc_dwu_avail
dwu_dnc_ready
PEP Controller Unit
Dotline Writer Unit

FIG. 254

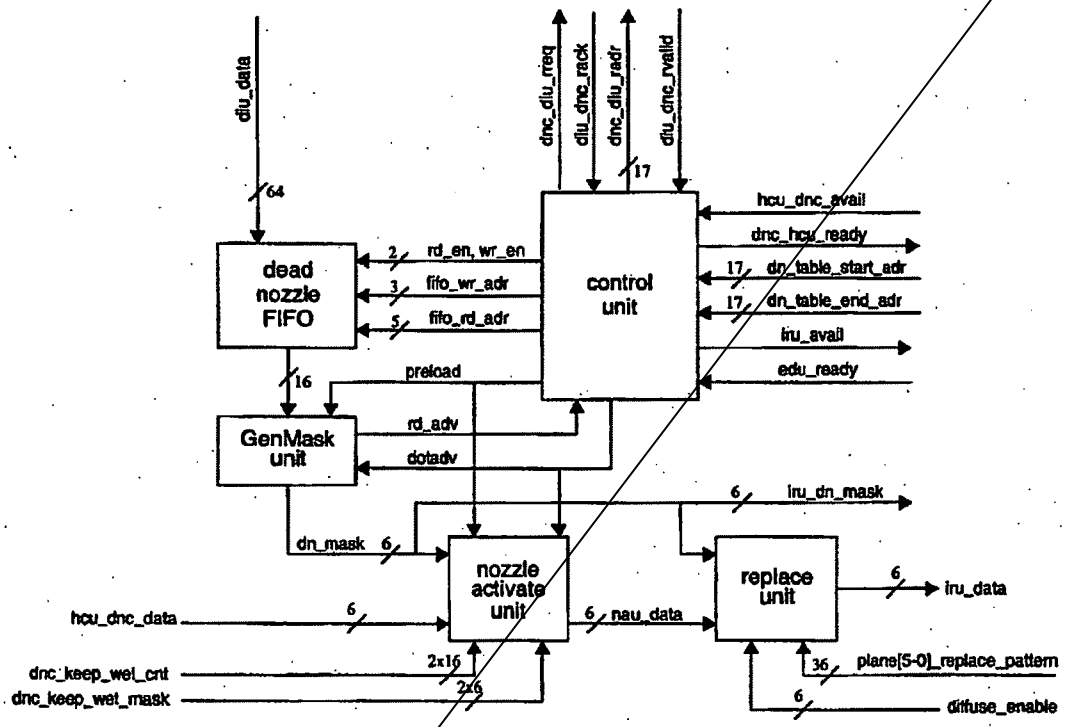

diu_data
64
dnc_diu_rreq
diu_dnc_rack
dnc_diu_radr
diu_dnc_rvalid
17
hcu_dnc_avail
dnc_hcu_ready
dn_table_start_adr
dn_table_end_adr
iru_avail
edu_ready
dead nozzle FIFO
2
rd_en, wr_en
3
fifo_wr_adr
5
fifo_rd_adr
control unit
16
preload
GenMask unit
rd_adv
dotadv
6
iru_dn_mask
dn_mask
6
nozzle activate unit
replace unit
6
iru_data
hcu_dnc_data
6
6
nau_data
2x16
dnc_keep_wet_cnt
2x6
dnc_keep_wet_mask
36
plane[5-0]_replace_pattern
6
diffuse_enable

FIG. 255

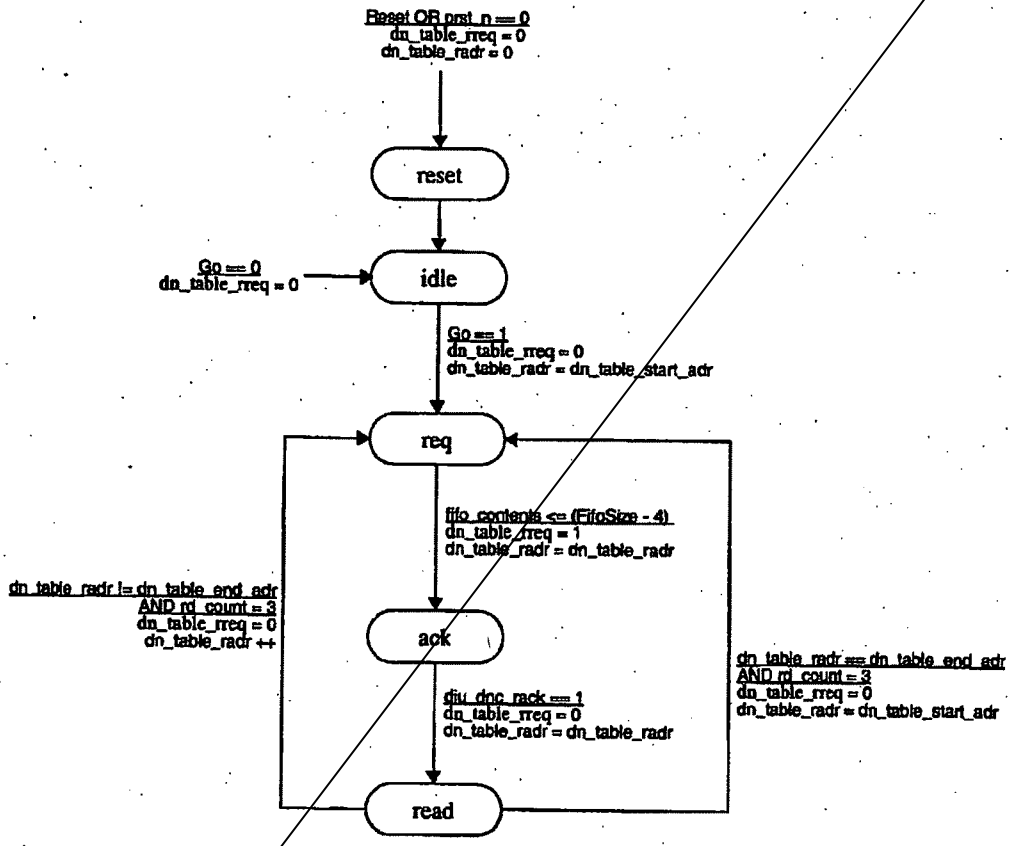

Reset OR prst_n == 0
dn_table_rreq = 0
dn_table_radr = 0
reset
Go == 0
dn_table_rreq = 0
idle
Go == 1
dn_table_rreq = 0
dn_table_radr = dn_table_start_adr
req
fifo_contents <= (FifoSize - 4)
dn_table_rreq = 1
dn_table_radr = dn_table_radr
dn_table_radr != dn_table_end_adr
AND rd_count = 3
dn_table_rreq = 0
dn_table_radr ++
ack
dn_table_radr == dn_table_end_adr
AND rd_count = 3
dn_table_rreq = 0
dn_table_radr = dn_table_start_adr
dlu_dnc_rack == 1
dn_table_rreq = 0
dn_table_radr = dn_table_radr
read

FIG. 256

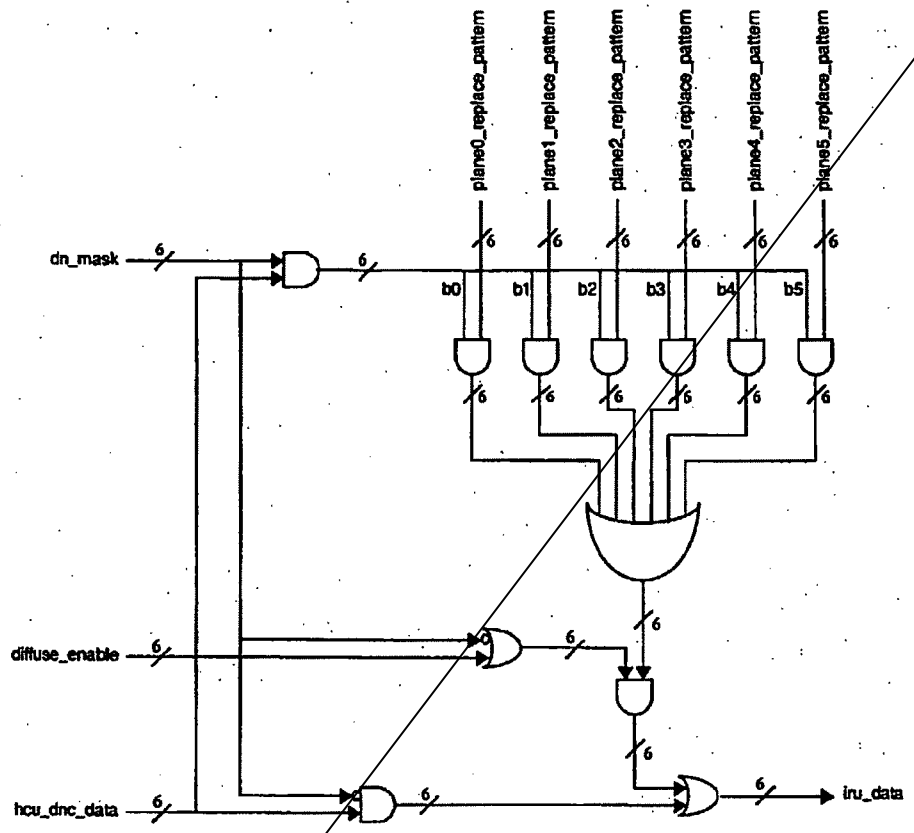


FIG. 257

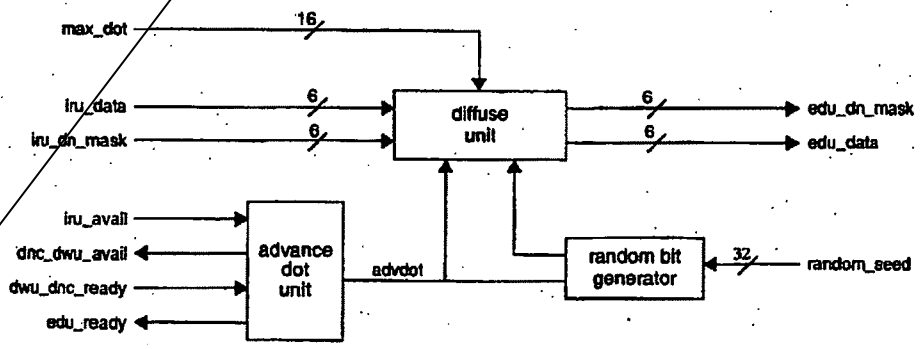


FIG. 258

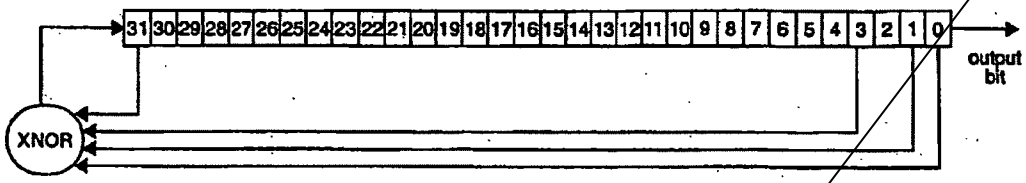


FIG. 259

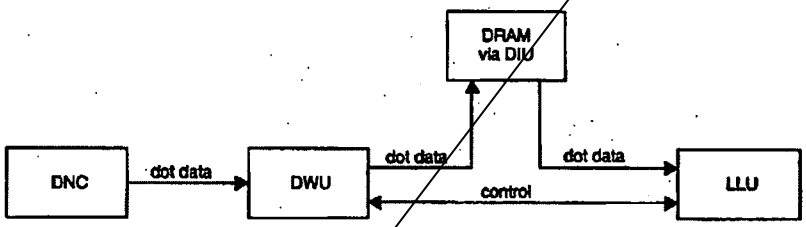
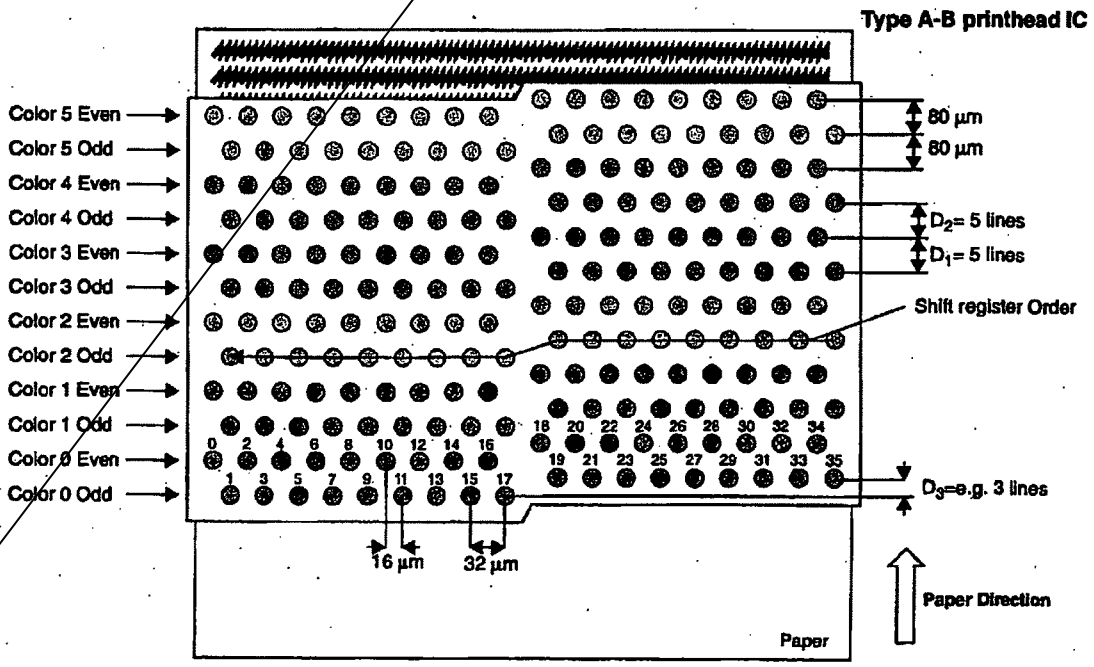


FIG. 260



Note: Paper passes under printhead

FIG. 261

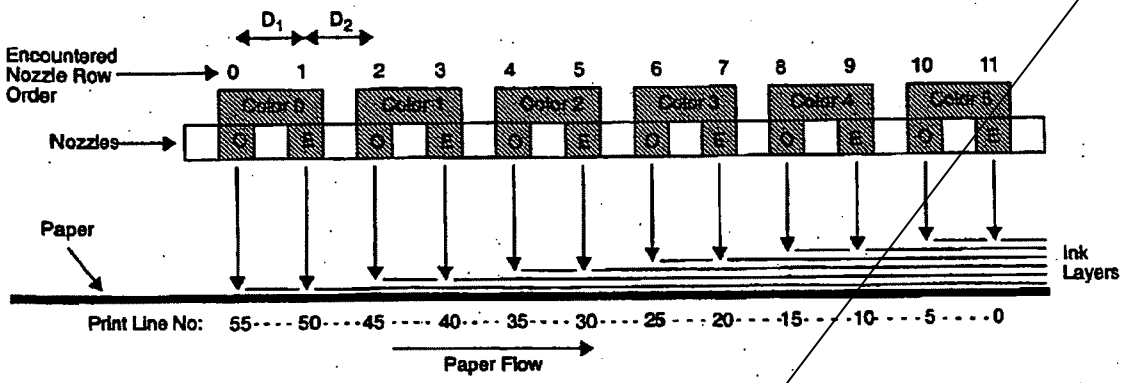


FIG. 262

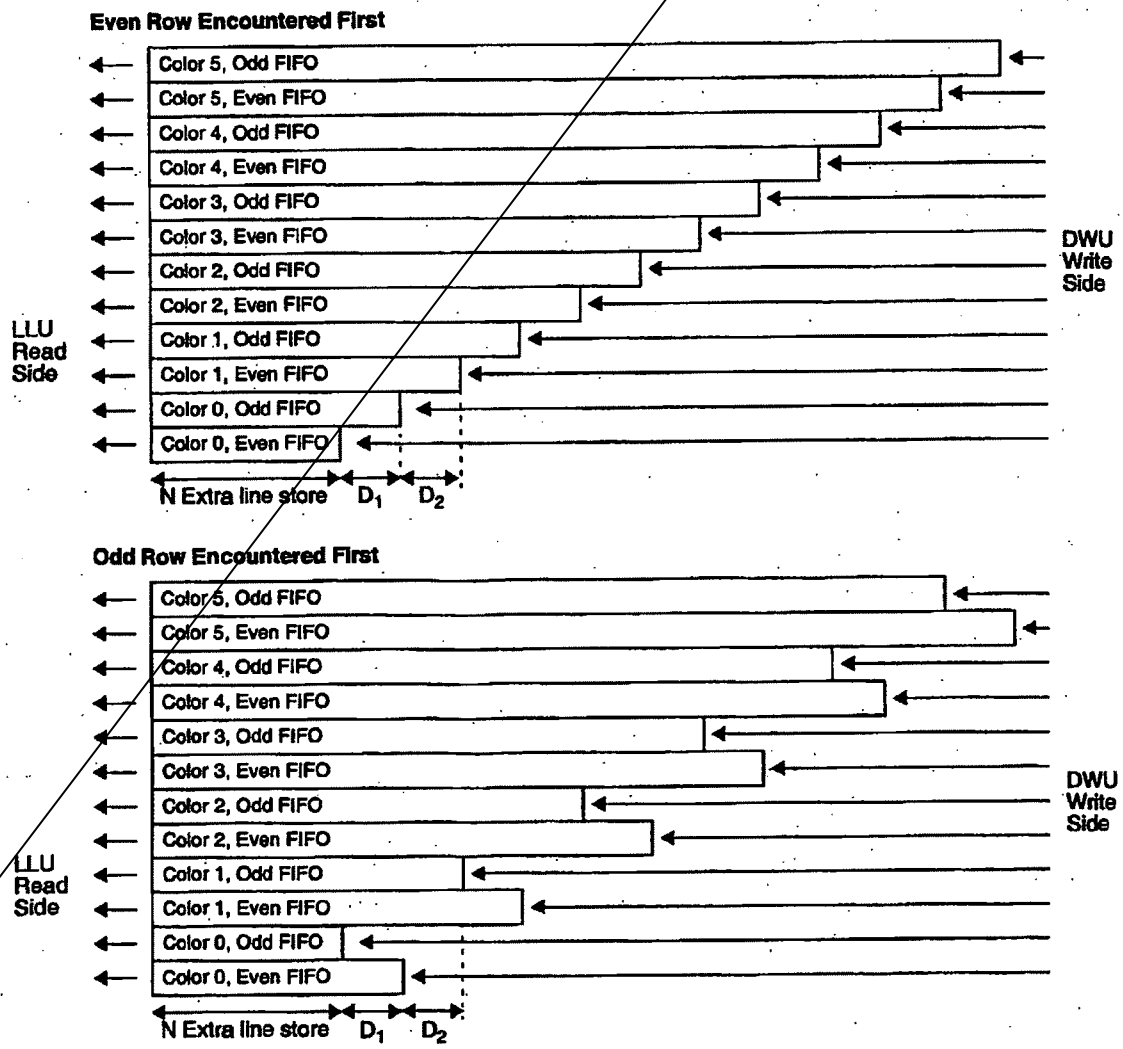


FIG. 263

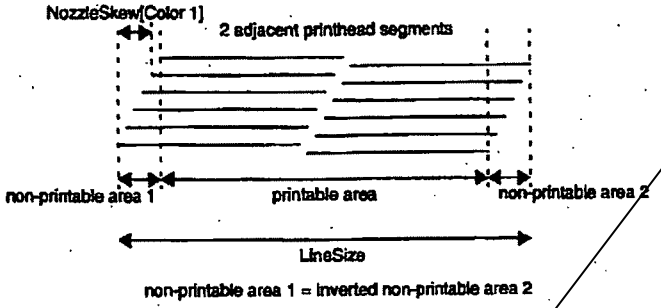


FIG. 264

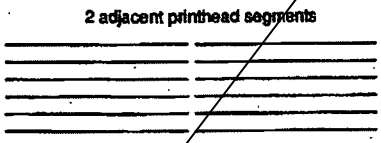


FIG. 265

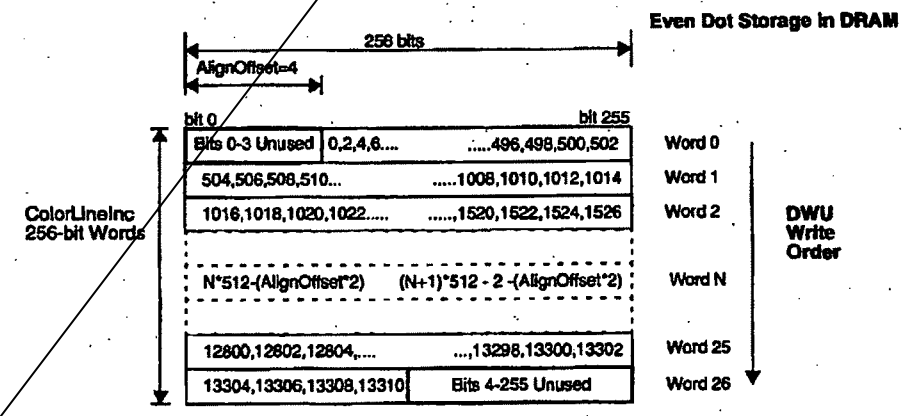


FIG. 266

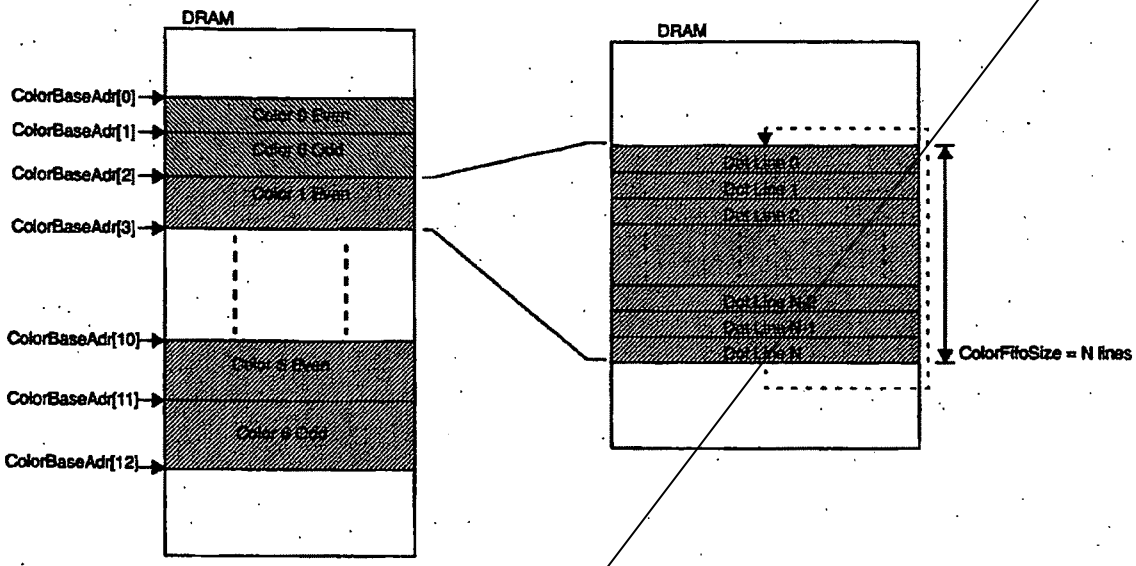

DRAM
ColorBaseAdr[0]
ColorBaseAdr[1]
ColorBaseAdr[2]
ColorBaseAdr[3]
ColorBaseAdr[10]
ColorBaseAdr[11]
ColorBaseAdr[12]
Color 0 Even
Color 0 Odd
Color 1 Even
Color 9 Even
Color 9 Odd
DRAM
ColorFifoSize = N lines

FIG. 267

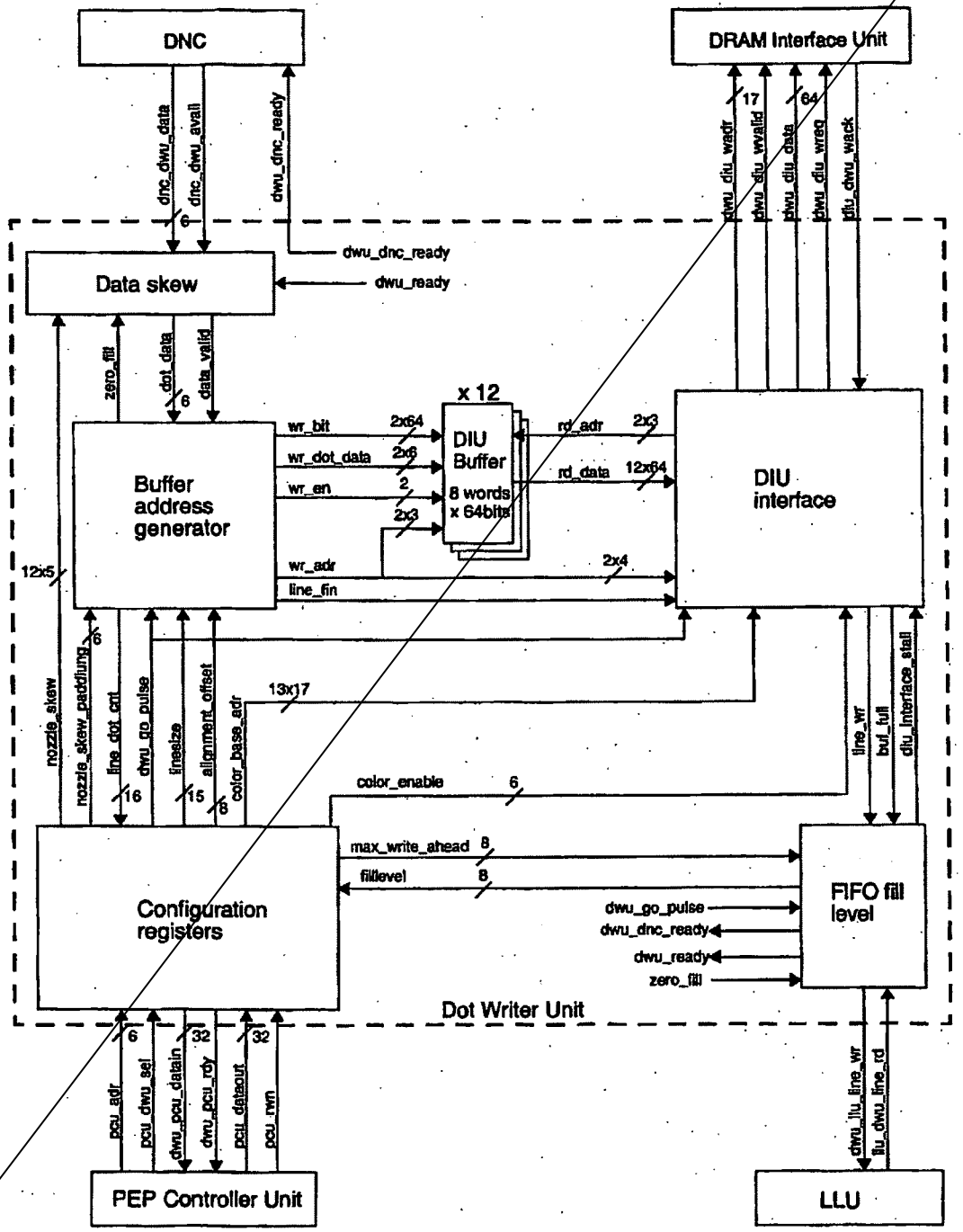

DNC
DRAM Interface Unit
dnc_dwu_data
dnc_dwu_avail
dwu_dnc_ready
dwu_diu_wadr
dwu_diu_wvalid
dwu_diu_data
dwu_diu_wreq
diu_dwu_wack
17
64
6
dwu_dnc_ready
Data skew
dwu_ready
zero_fill
dot_data
data_valid
x 12
wr_bit
wr_dot_data
wr_en
wr_adr
line_fin
2x64
2x6
2
2x3
2x4
DIU Buffer
8 words x 64bits
rd_adr
rd_data
12x64
Buffer address generator
DIU interface
12x5
6
nozzle_skew
nozzle_skew_padding
line_dot_cnt
dwu_go_pulse
fineize
alignment_offset
color_base_adr
13x17
16
15
8
color_enable
6
line_wr
buf_full
diu_interface_stall
max_write_ahead
fillevel
8
8
Configuration registers
FIFO fill level
dwu_go_pulse
dwu_dnc_ready
dwu_ready
zero_fill
Dot Writer Unit
6
32
32
pcu_adr
pcu_dwu_sel
dwu_pcu_datain
dwu_pcu_rdy
pcu_dataout
pcu_rwn
dwu_llu_line_wr
llu_dwu_line_rd
PEP Controller Unit
LLU

FIG. 268

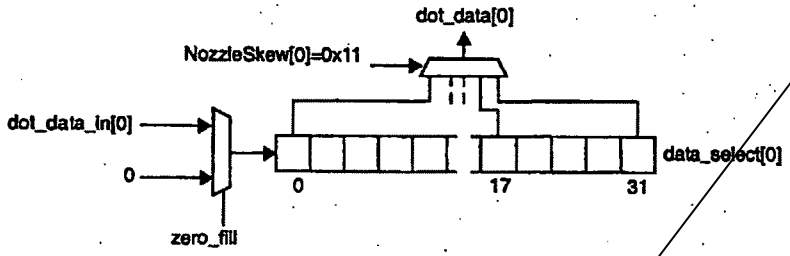


FIG. 269

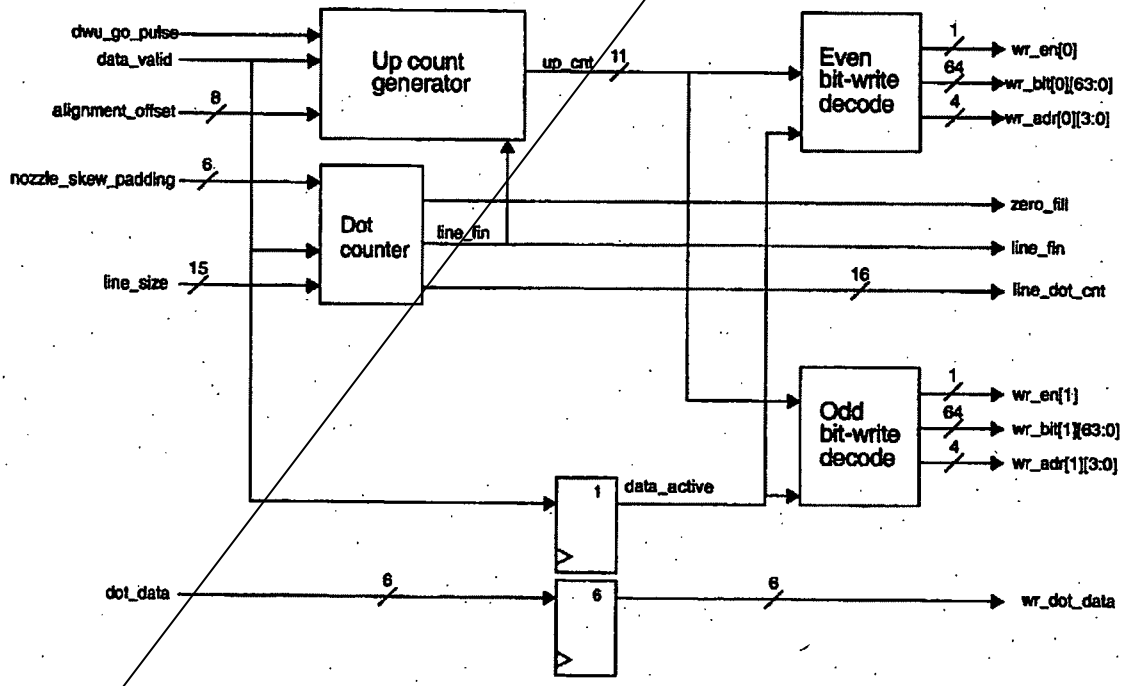


FIG. 270

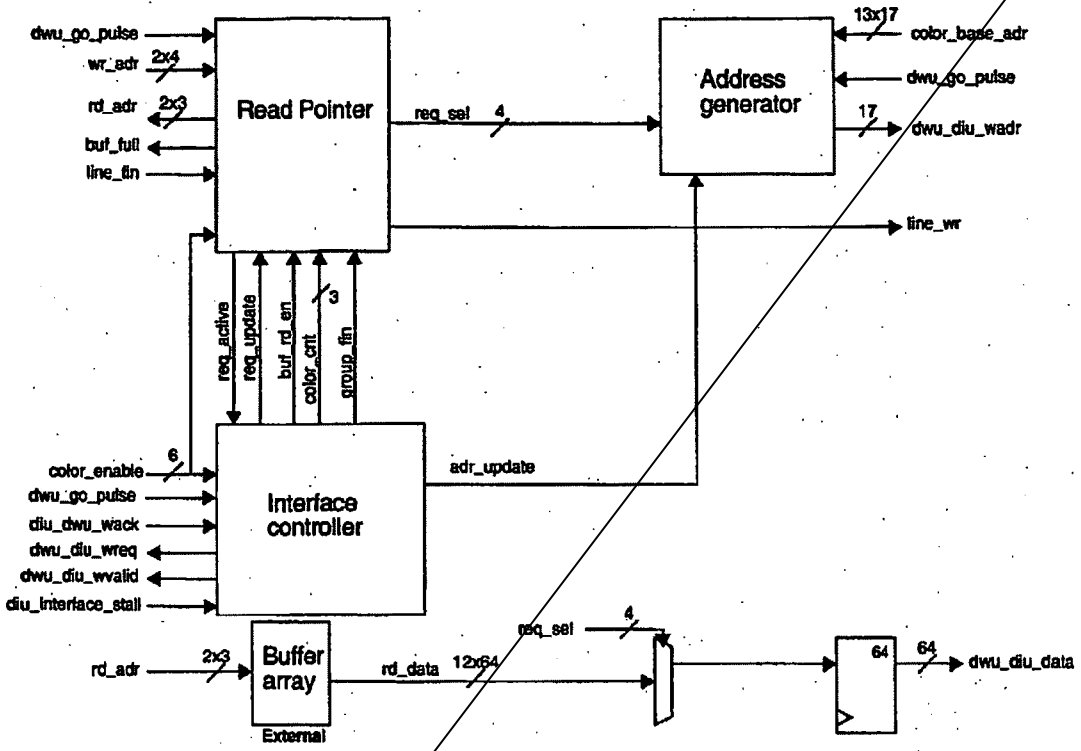
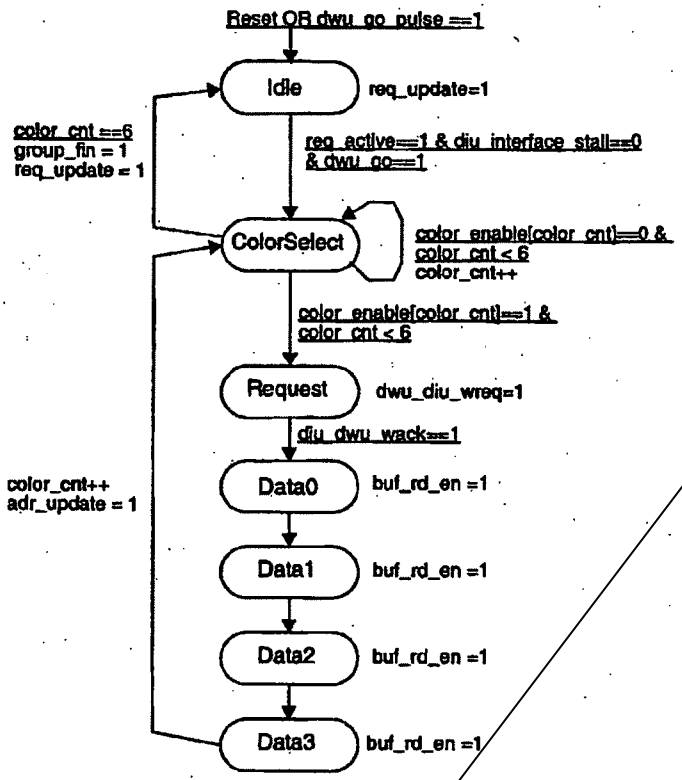

dwu_go_pulse
wr_adr
2x4
rd_adr
2x3
buf_full
line_fin
Read Pointer
req_sel
4
Address generator
13x17
color_base_adr
dwu_go_pulse
17
dwu_diu_wadr
line_wr
req_active
req_update
buf_rd_en
color_cnt
3
group_fin
color_enable
6
dwu_go_pulse
diu_dwu_wack
dwu_diu_wreq
dwu_diu_wvalid
diu_interface_stall
Interface controller
adr_update
rd_adr
2x3
Buffer array
External
rd_data
12x64
req_sel
4
64
64
dwu_diu_data

FIG. 271



Machine remains in same state by default
All outputs are zero unless otherwise stated

State Description:

Idle:	Idle state wait for active request
ColorSelect:	Select the color to update before requesting to DIU
Request:	Request issued wait for acknowledge
Data0:	Data word 0 transfer
Data1:	Data word 1 transfer
Data2:	Data word 2 transfer
Data3:	Data word 3 transfer

FIG. 272

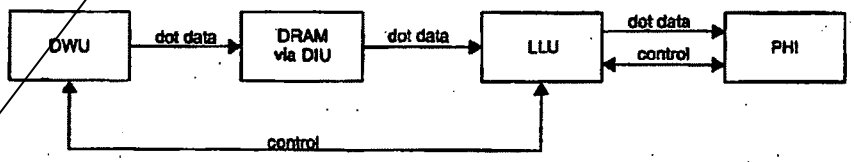


FIG. 273

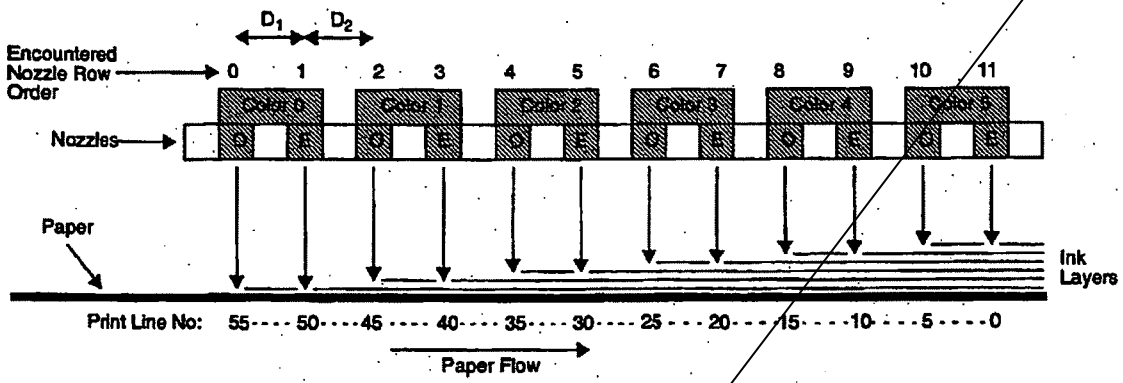


FIG. 274

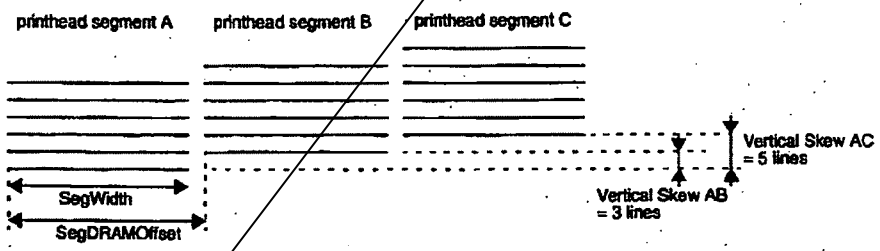


FIG. 275

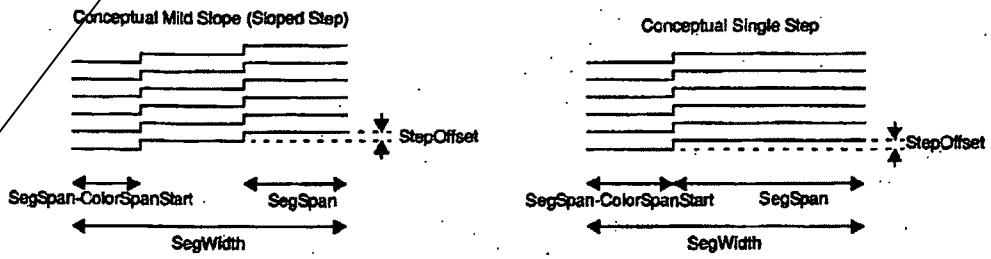


FIG. 276

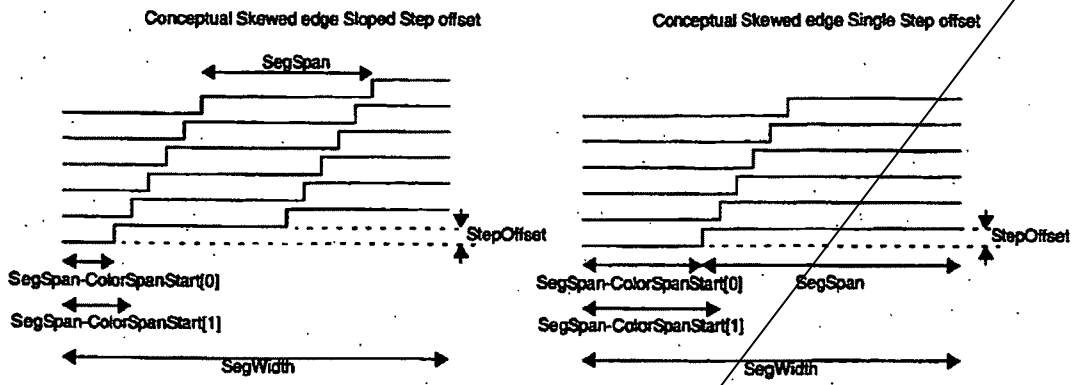


FIG. 277

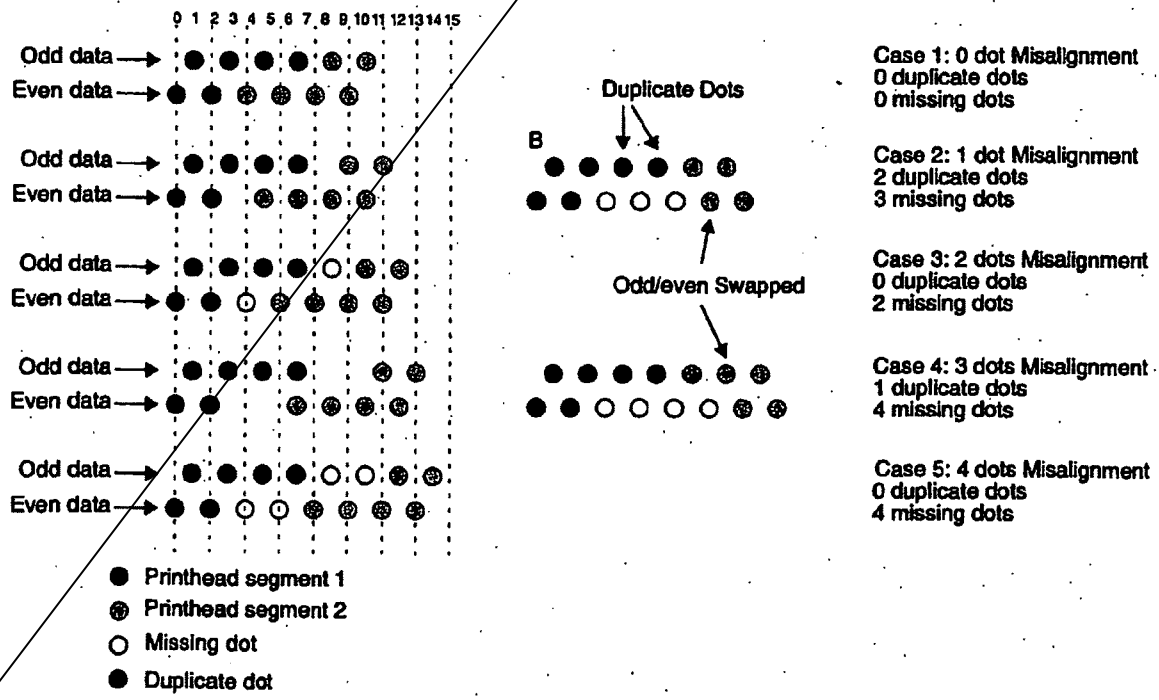


FIG. 278

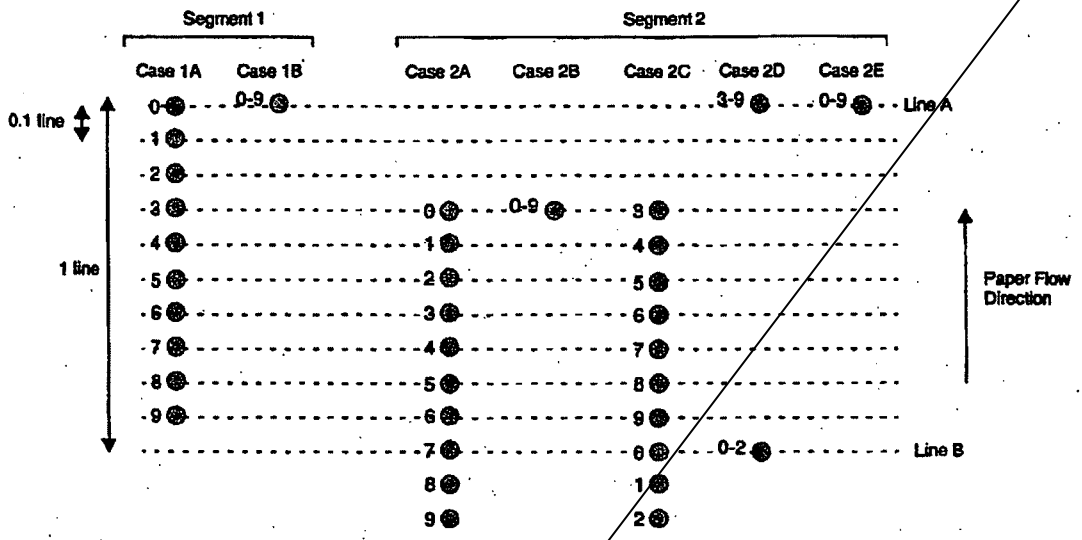


FIG. 279

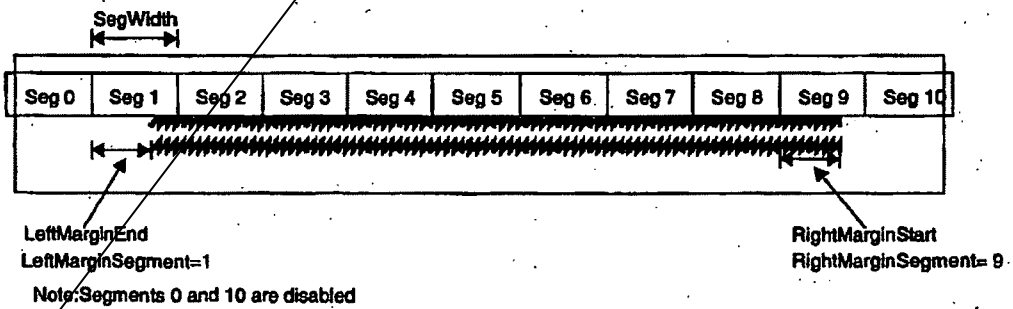


FIG. 280

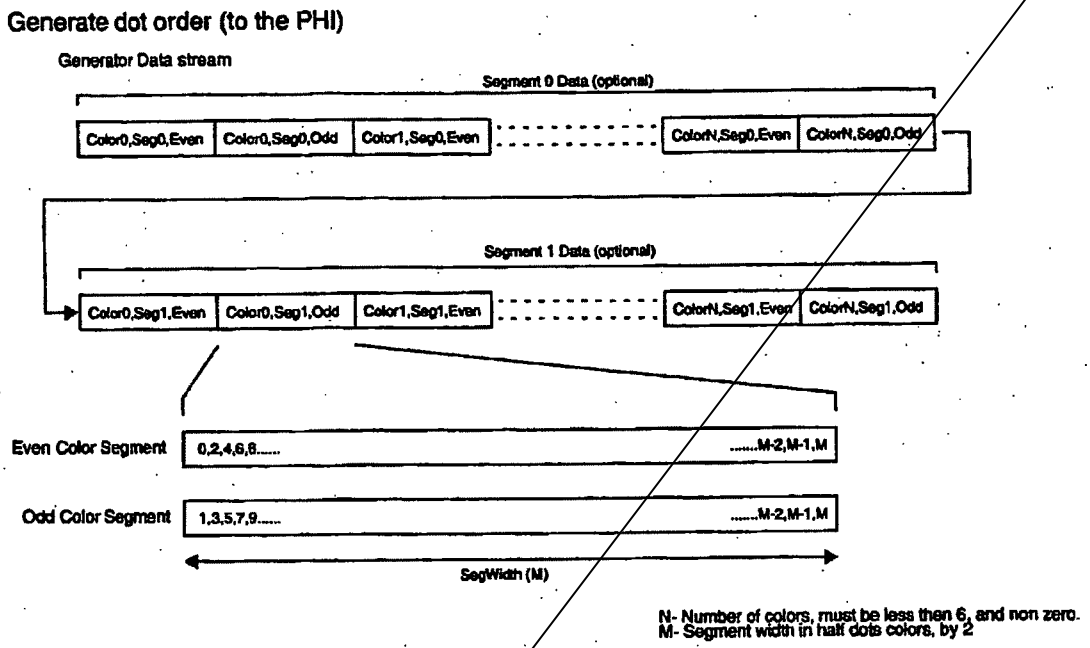


FIG. 281

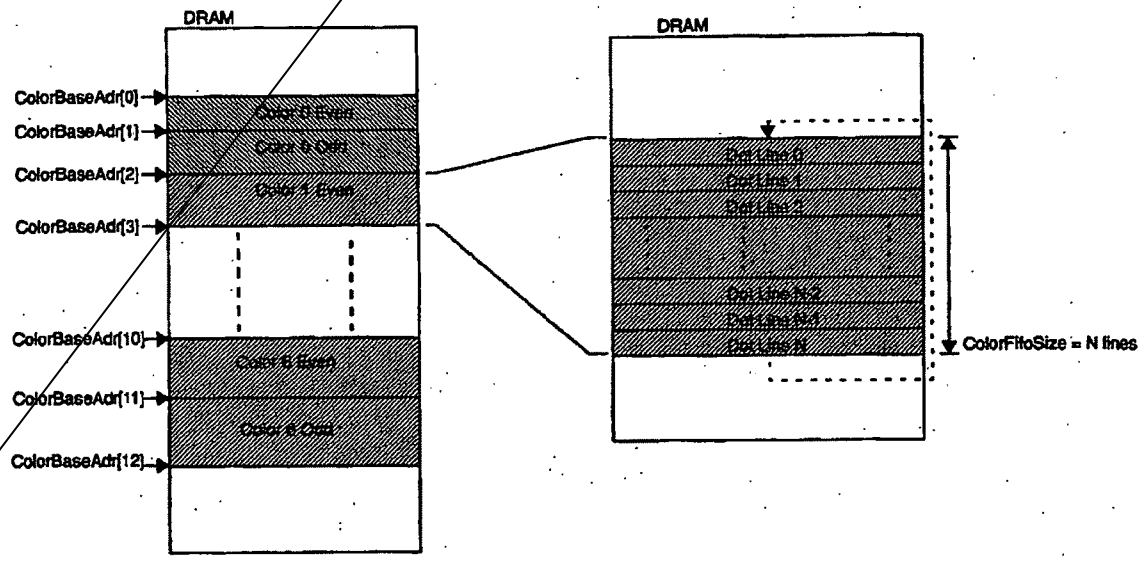
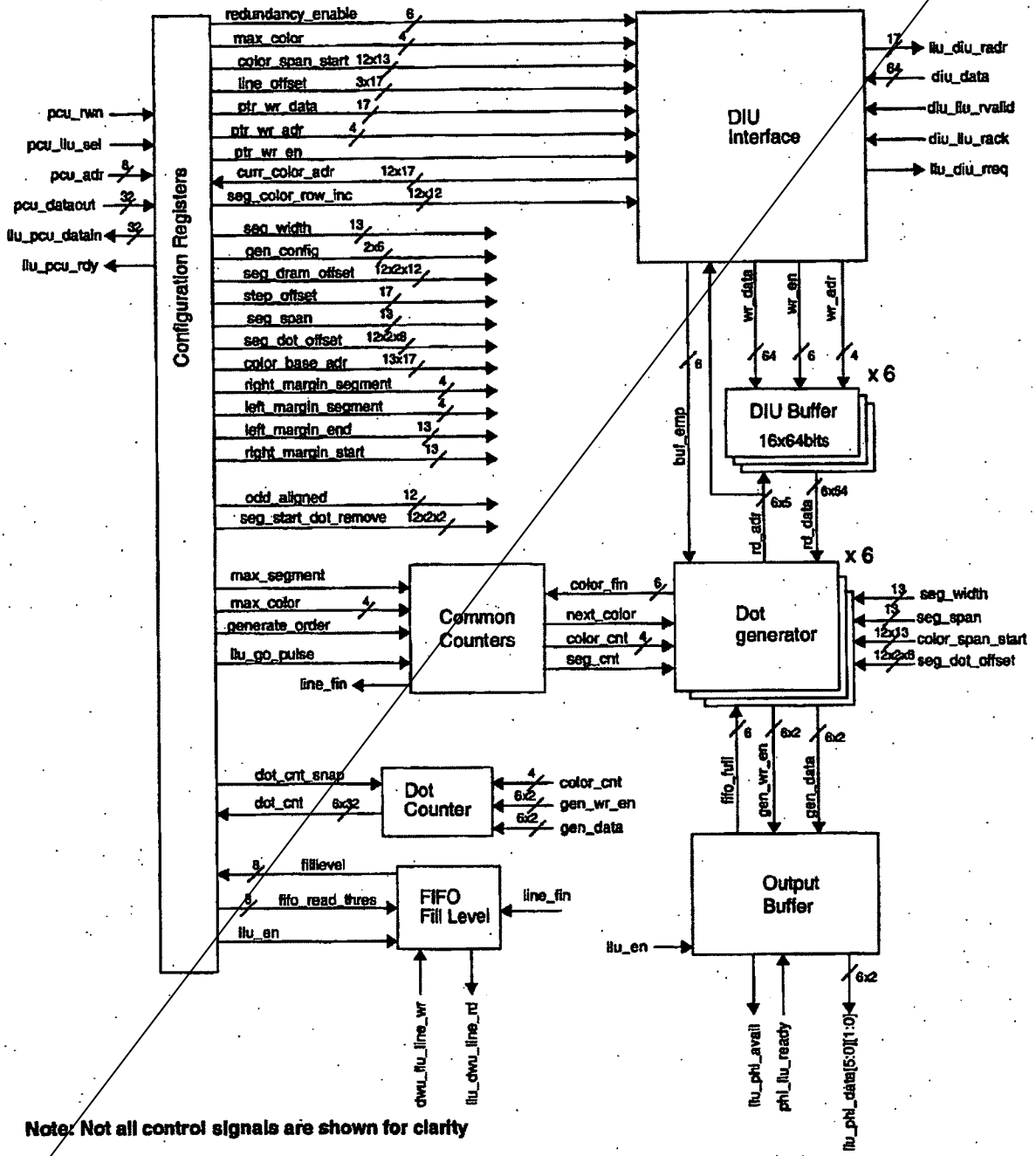


FIG. 282



Note: Not all control signals are shown for clarity

FIG. 283

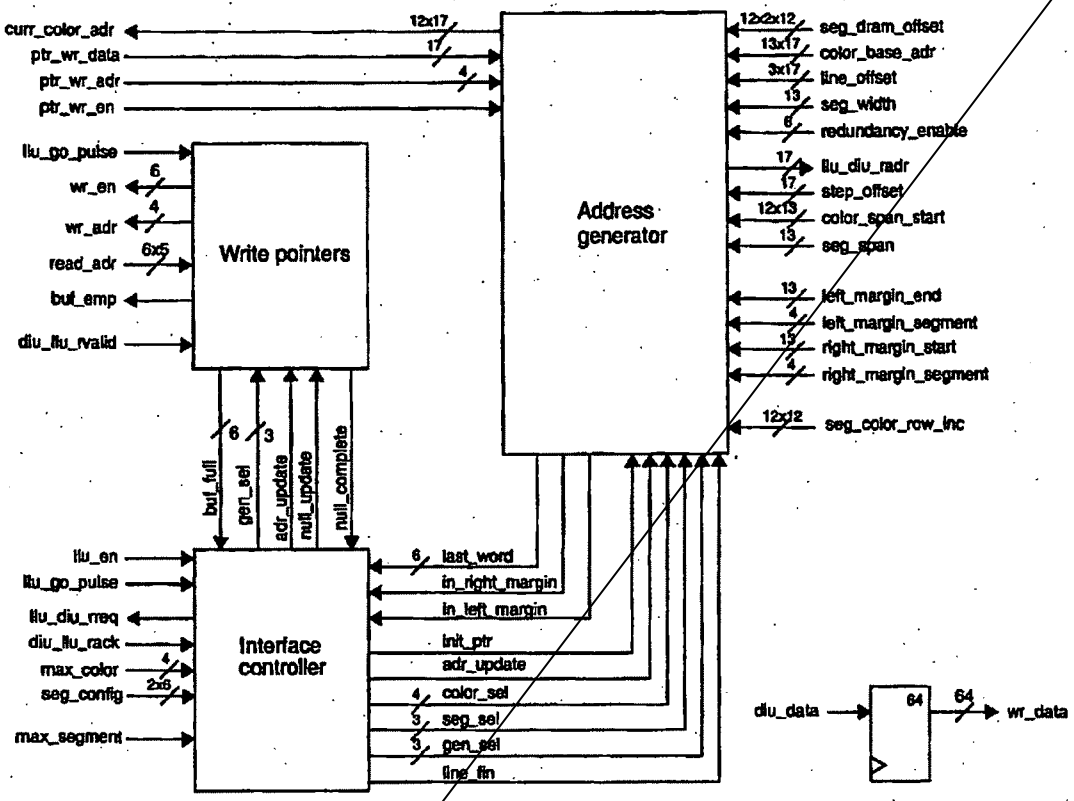
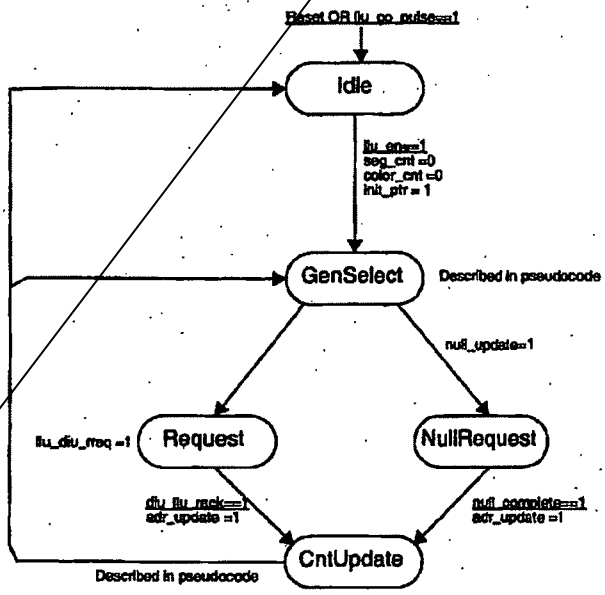


FIG. 284



Machine remains in same state by default
All outputs are zero unless otherwise stated

State Description:

Idle:	Idle state wait for active request
GenSelect:	Select the correct generator to update before requesting to DIU
Request:	Request issued wait for acknowledge
NullRequest	Request that buffer be filled with null data, wait for completion
CntUpdate	Update the address pointers

FIG. 285

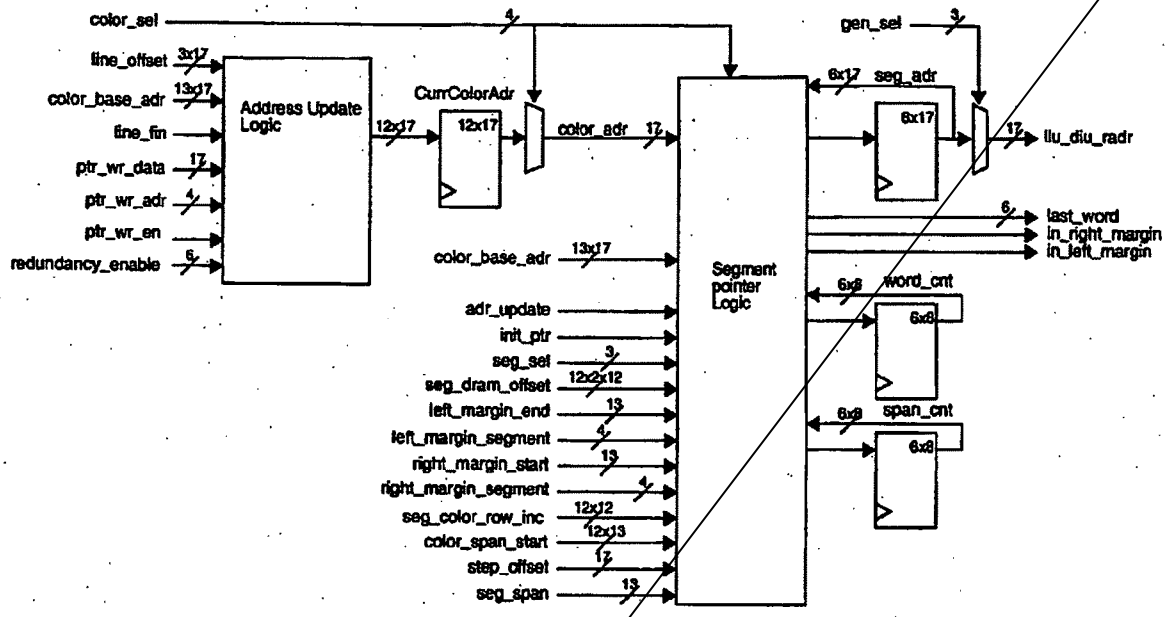


FIG. 286

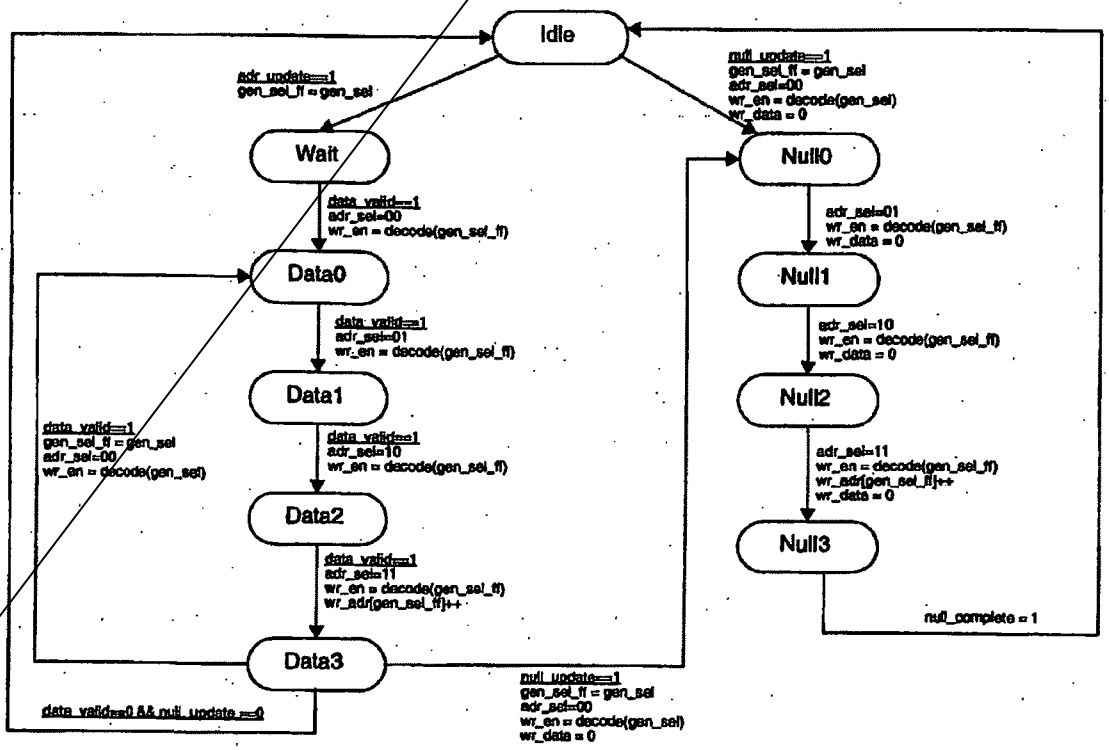


FIG. 287

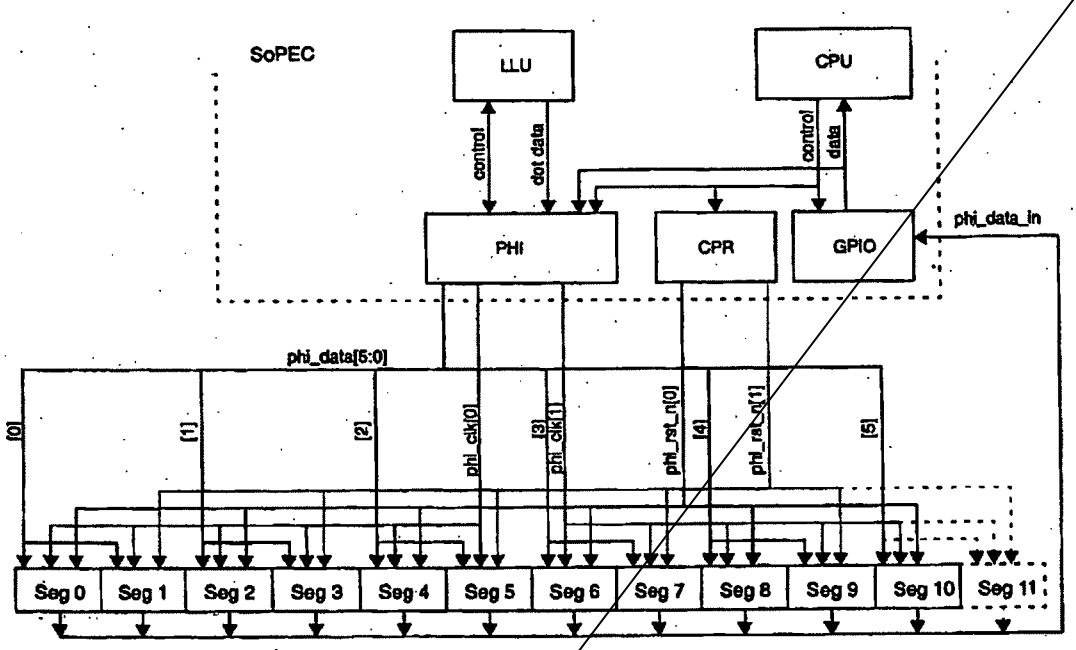


FIG. 288

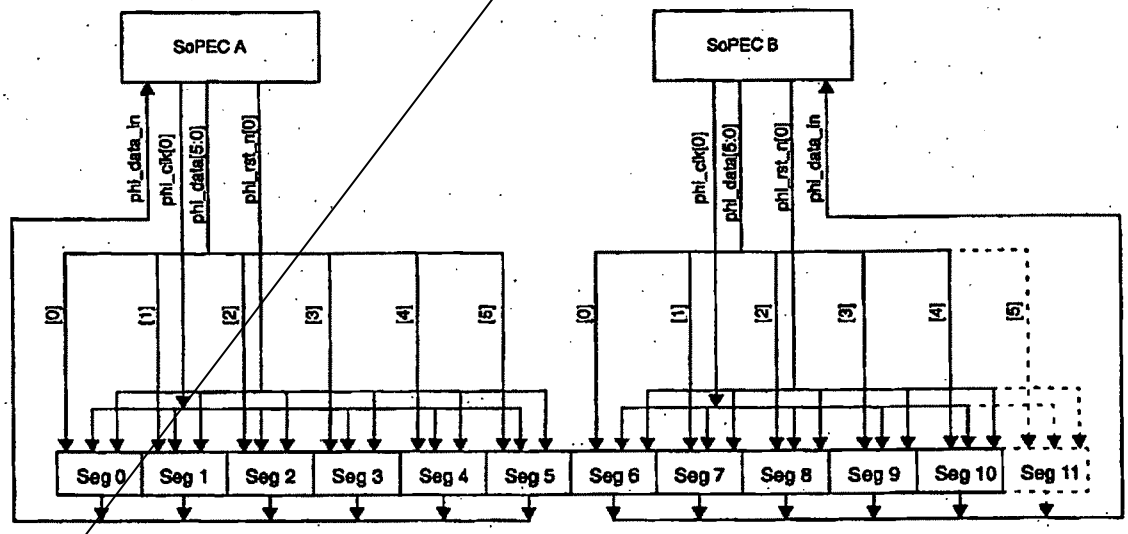


FIG. 289

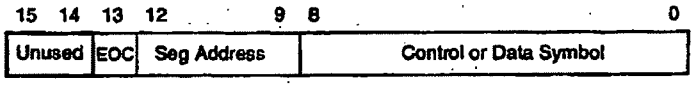


FIG. 290

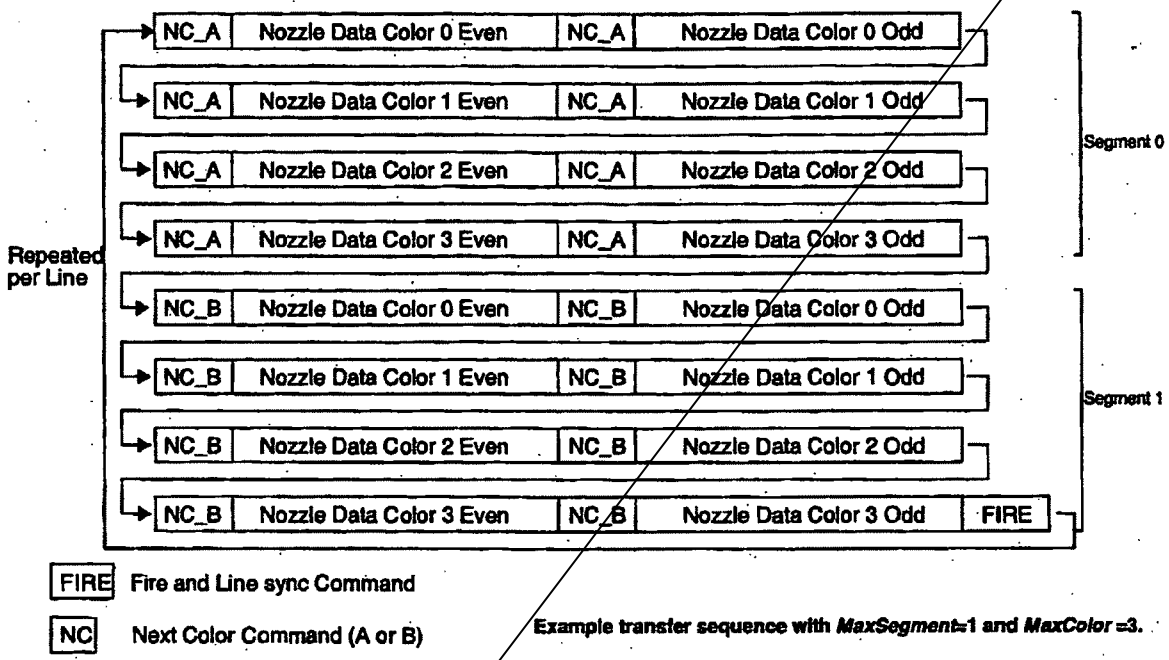

NC_A
Nozzle Data Color 0 Even
NC_A
Nozzle Data Color 0 Odd
NC_A
Nozzle Data Color 1 Even
NC_A
Nozzle Data Color 1 Odd
NC_A
Nozzle Data Color 2 Even
NC_A
Nozzle Data Color 2 Odd
NC_A
Nozzle Data Color 3 Even
NC_A
Nozzle Data Color 3 Odd
Segment 0
Repeated per Line
NC_B
Nozzle Data Color 0 Even
NC_B
Nozzle Data Color 0 Odd
NC_B
Nozzle Data Color 1 Even
NC_B
Nozzle Data Color 1 Odd
NC_B
Nozzle Data Color 2 Even
NC_B
Nozzle Data Color 2 Odd
NC_B
Nozzle Data Color 3 Even
NC_B
Nozzle Data Color 3 Odd
FIRE
Segment 1
FIRE
Fire and Line sync Command
NC
Next Color Command (A or B)
Example transfer sequence with MaxSegment=1 and MaxColor =3.

FIG. 291

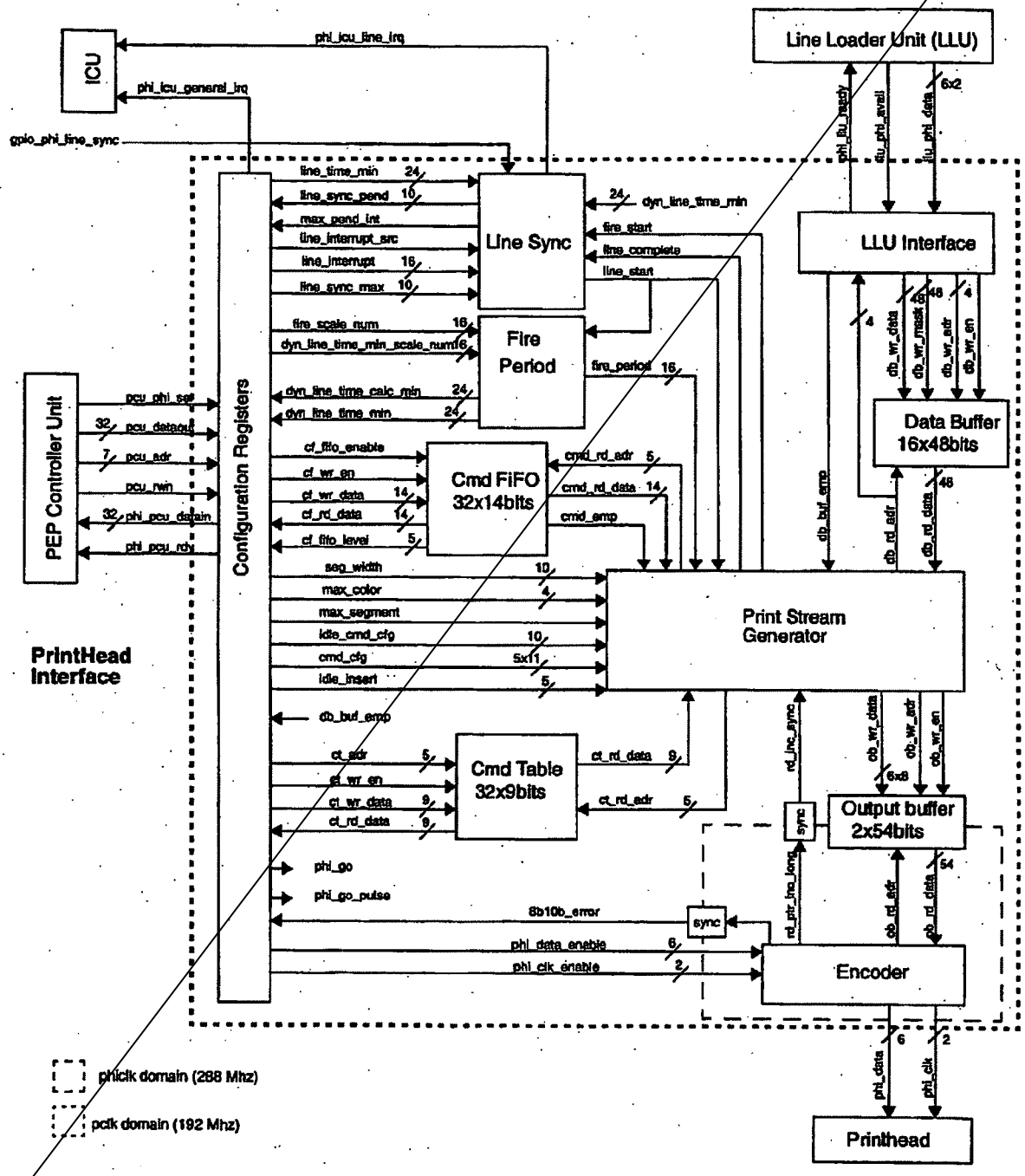

ICU
phi_icu_line_irq
phi_icu_general_irq
gpio_phi_line_sync
Line Loader Unit (LLU)
phi_llu_ready
llu_phi_avail
llu_phi_data
6x2
LLU Interface
Line Sync
Fire Period
line_time_min
line_sync_pend
max_pend_int
line_interrupt_src
line_interrupt
line_sync_max
fire_scale_num
dyn_line_time_min_scale_num
dyn_line_time_calc_min
dyn_line_time_min
dyn_line_time_min
fire_start
fire_complete
line_start
fire_period
db_wr_data
db_wr_mask
db_wr_adr
db_wr_en
Data Buffer 16x48bits
PEP Controller Unit
pcu_phi_sel
pcu_dataout
pcu_adr
pcu_rwn
phi_pcu_datain
phi_pcu_rdy
Configuration Registers
cf_fifo_enable
cf_wr_en
cf_wr_data
cf_rd_data
cf_fifo_level
Cmd FiFO 32x14bits
cmd_rd_adr
cmd_rd_data
cmd_emp
db_buf_emp
db_rd_adr
db_rd_data
seg_width
max_color
max_segment
idle_cmd_cfg
cmd_cfg
idle_insert
Print Stream Generator
PrintHead Interface
db_buf_emp
ct_adr
ct_wr_en
ct_wr_data
ct_rd_data
Cmd Table 32x9bits
ct_rd_data
ct_rd_adr
rd_inc_sync
sync
Output buffer 2x54bits
phi_go
phi_go_pulse
8b10b_error
phi_data_enable
phi_clk_enable
rd_ptr_inc_long
Encoder
phi_data
phi_clk
Printhead
phiclk domain (288 Mhz)
pclk domain (192 Mhz)

FIG. 292

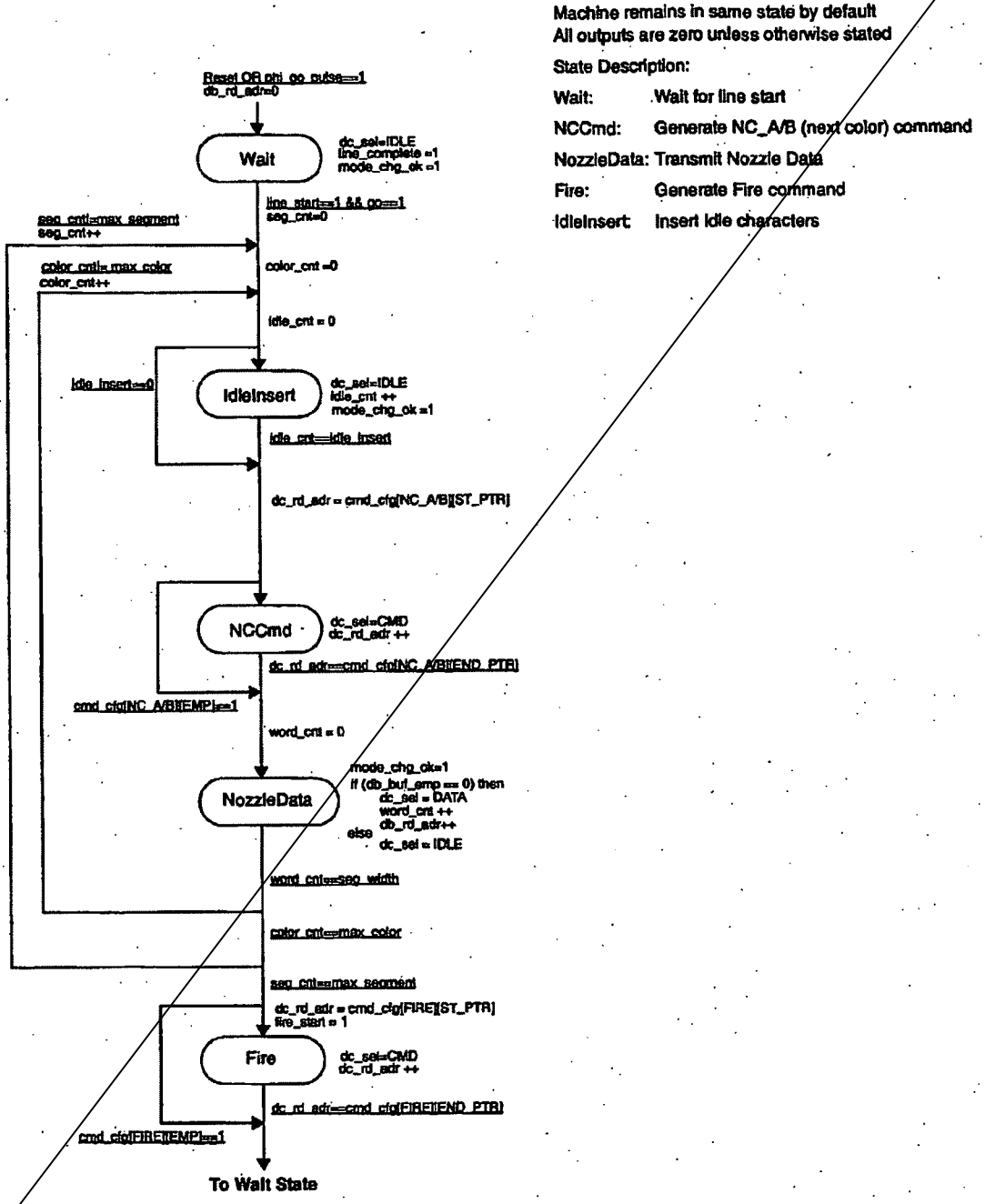

Machine remains in same state by default
All outputs are zero unless otherwise stated
State Description:
Wait: Wait for line start
NCCmd: Generate NC_A/B (next color) command
NozzleData: Transmit Nozzle Data
Fire: Generate Fire command
IdleInsert: Insert idle characters
Reset OR pbi_go_pulse==1
db_rd_adr=0
Wait
dc_sel=IDLE
line_complete =1
mode_chg_ok =1
line_start==1 && go==1
seg_cnt=0
seg_cnt=max_segment
seg_cnt++
color_cnt=max_color
color_cnt++
color_cnt =0
idle_cnt = 0
idle_insert==0
IdleInsert
dc_sel=IDLE
idle_cnt ++
mode_chg_ok =1
idle_cnt==idle_insert
dc_rd_adr = cmd_cfg[NC_A/B][ST_PTR]
NCCmd
dc_sel=CMD
dc_rd_adr ++
dc_rd_adr==cmd_cfg[NC_A/B][END_PTR]
cmd_cfg[NC_A/B][EMPTY]==1
word_cnt = 0
NozzleData
mode_chg_ok=1
if (db_buf_emp == 0) then
dc_sel = DATA
word_cnt ++
db_rd_adr++
else
dc_sel = IDLE
word_cnt==seg_width
color_cnt==max_color
seg_cnt==max_segment
dc_rd_adr = cmd_cfg[FIRE][ST_PTR]
fire_start = 1
Fire
dc_sel=CMD
dc_rd_adr ++
dc_rd_adr==cmd_cfg[FIRE][END_PTR]
cmd_cfg[FIRE][EMPTY]==1
To Wait State

FIG. 293

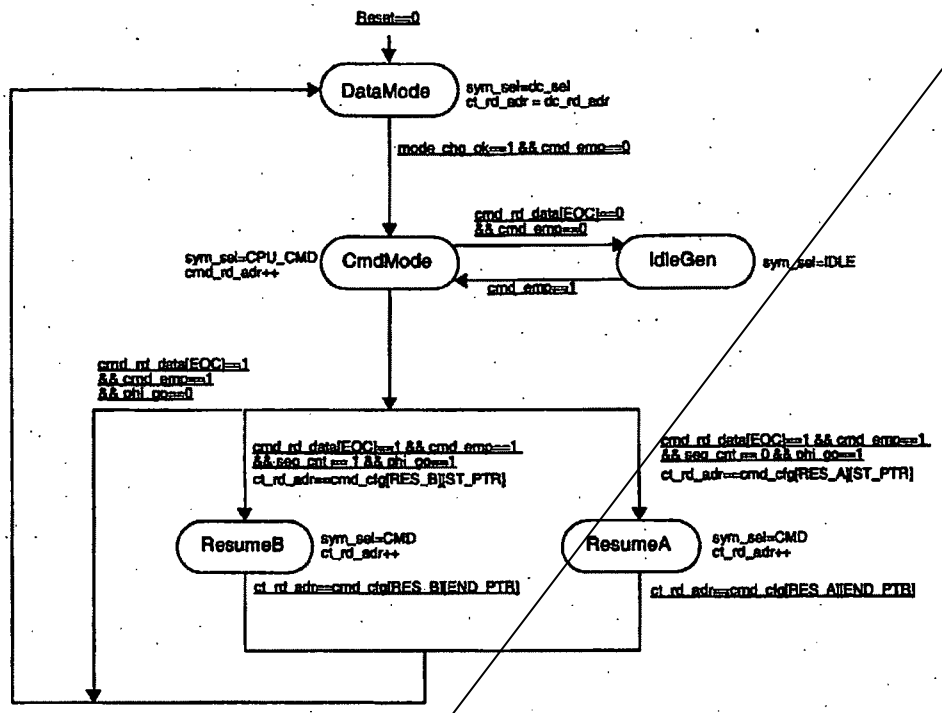


FIG. 294

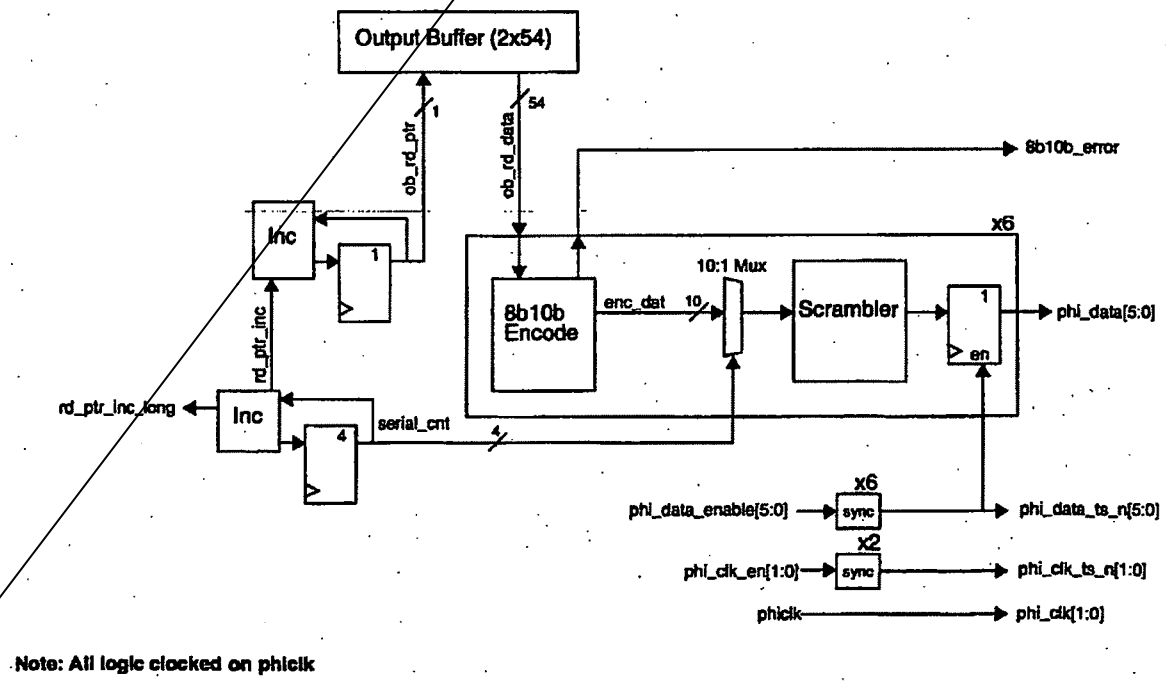


FIG. 295

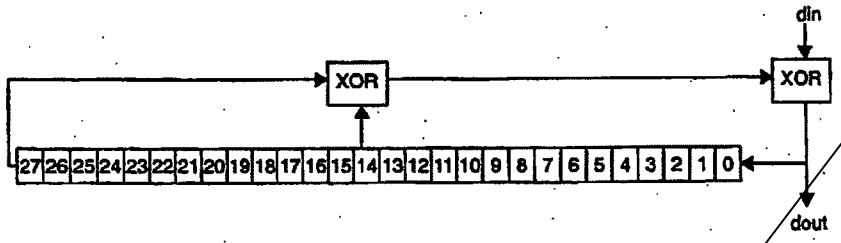


FIG. 296

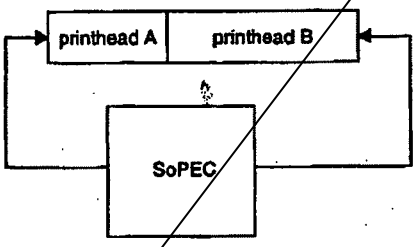


FIG. 297

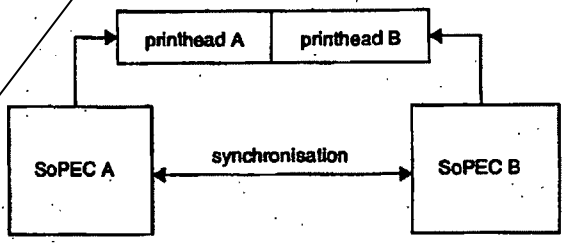


FIG. 298

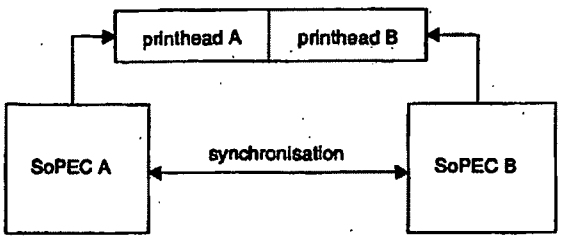


FIG. 299

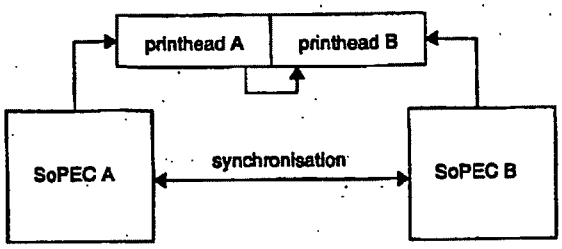


FIG. 300

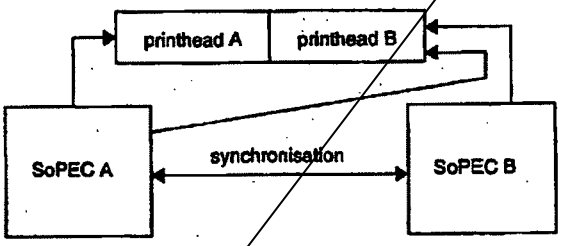


FIG. 301

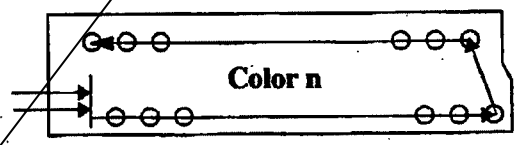


FIG. 302

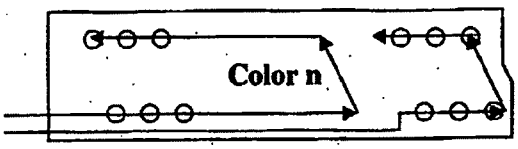


FIG. 303

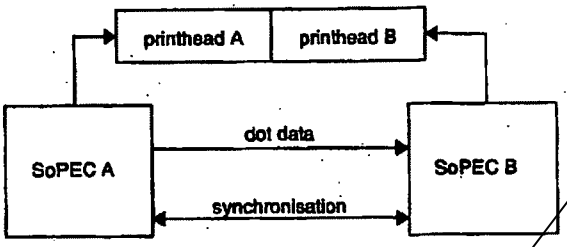


FIG. 304

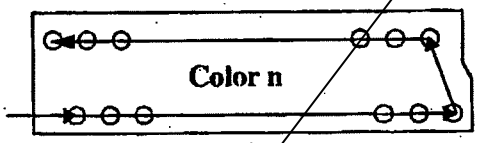


FIG. 305

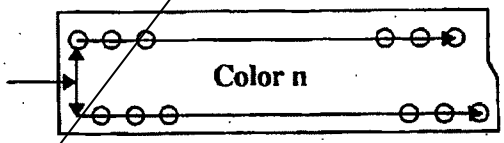


FIG. 306

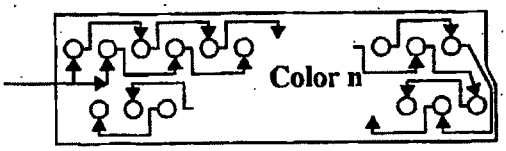


FIG. 307



FIG. 308

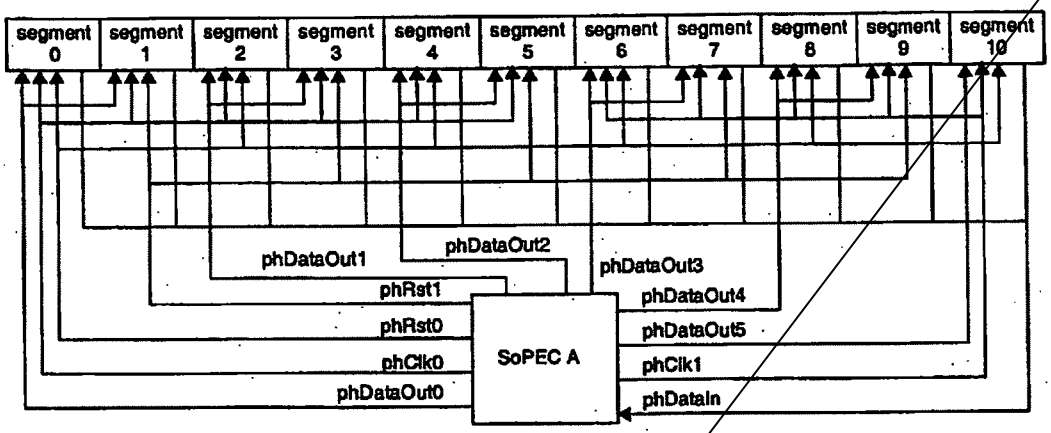


FIG. 309

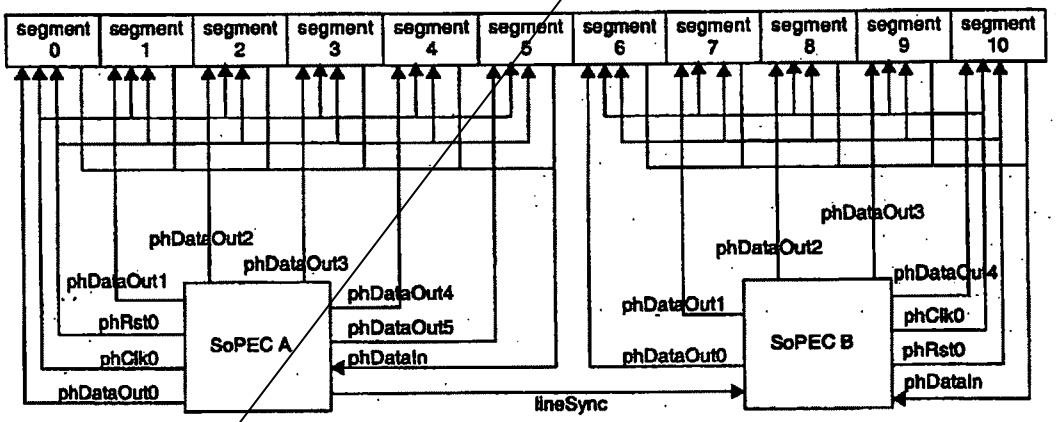


FIG. 310

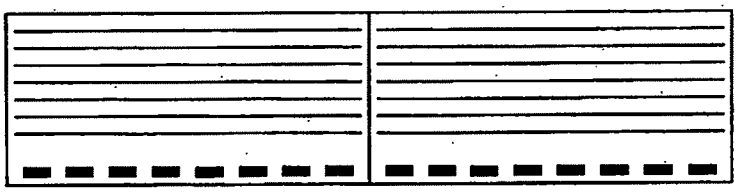


FIG. 311

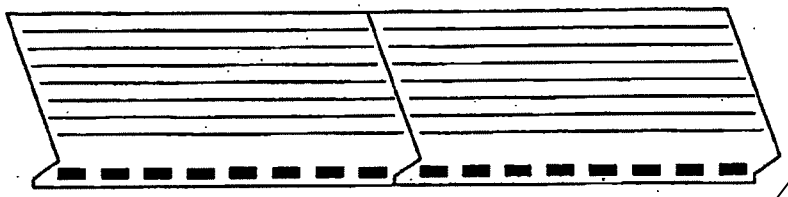


FIG. 312

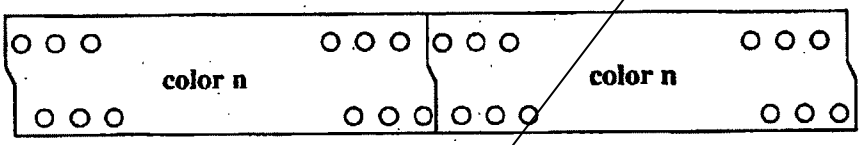


FIG. 313

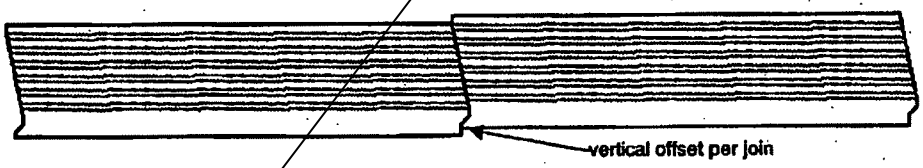


FIG. 314

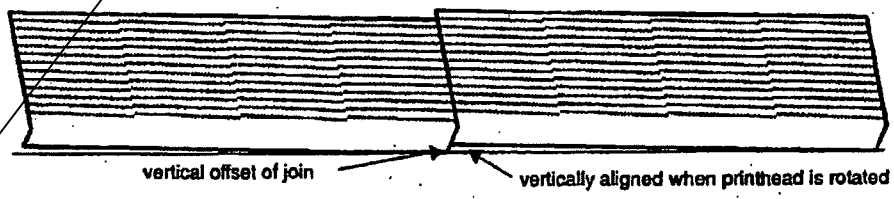


FIG. 315

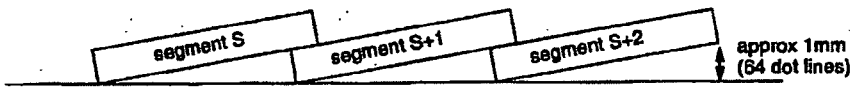


FIG. 316

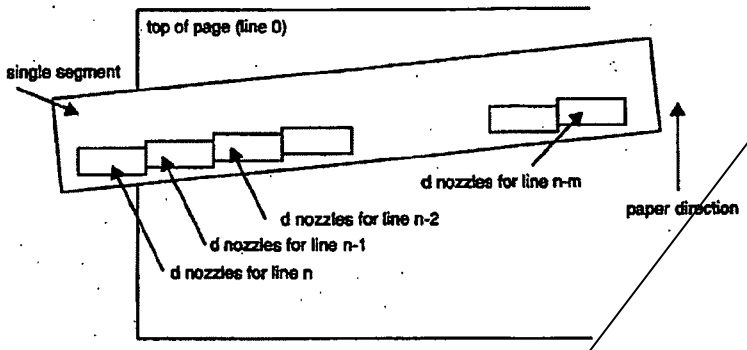


FIG. 317



FIG. 318

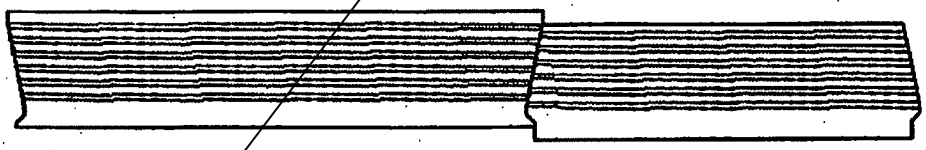


FIG. 319



FIG. 320

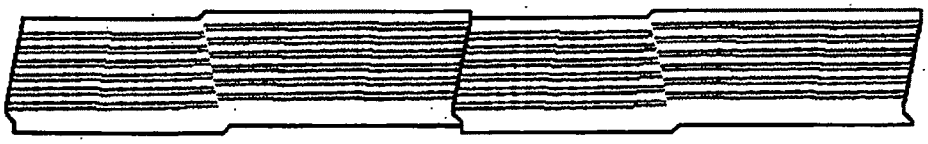


FIG. 321

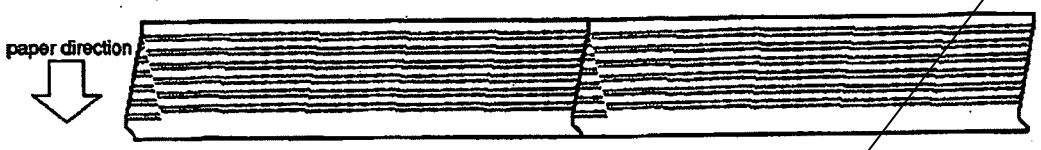


FIG. 322

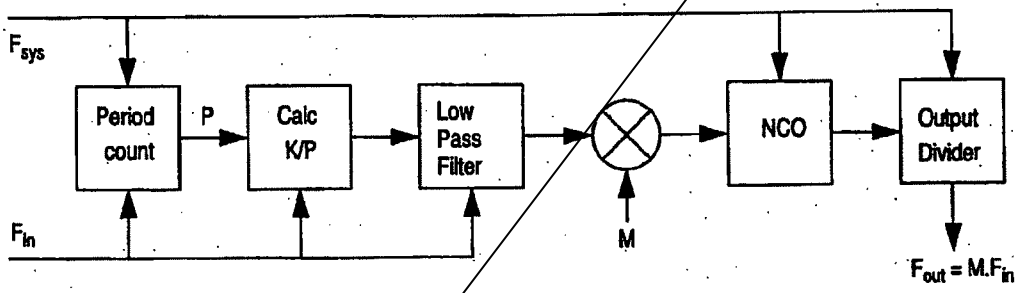


FIG. 323

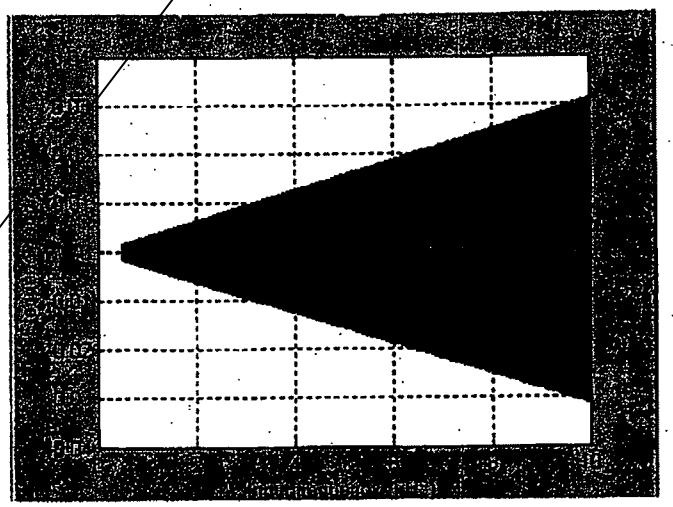


FIG. 324

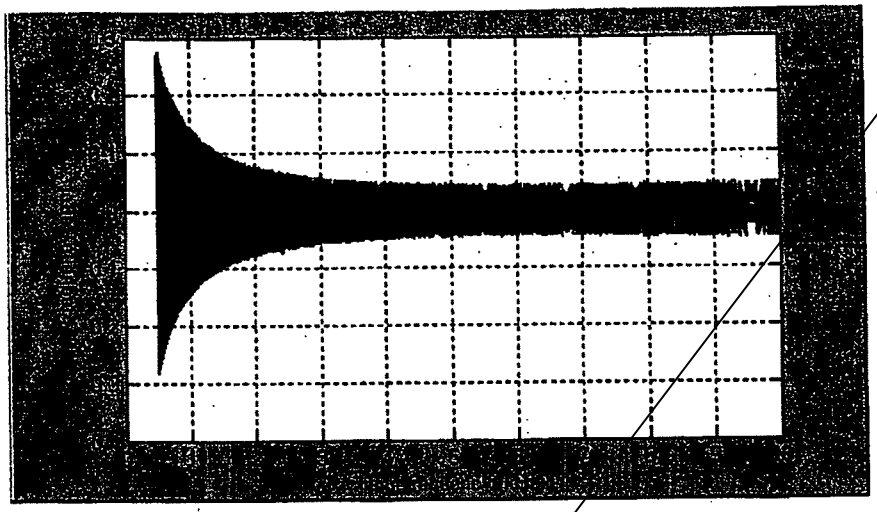


FIG. 325

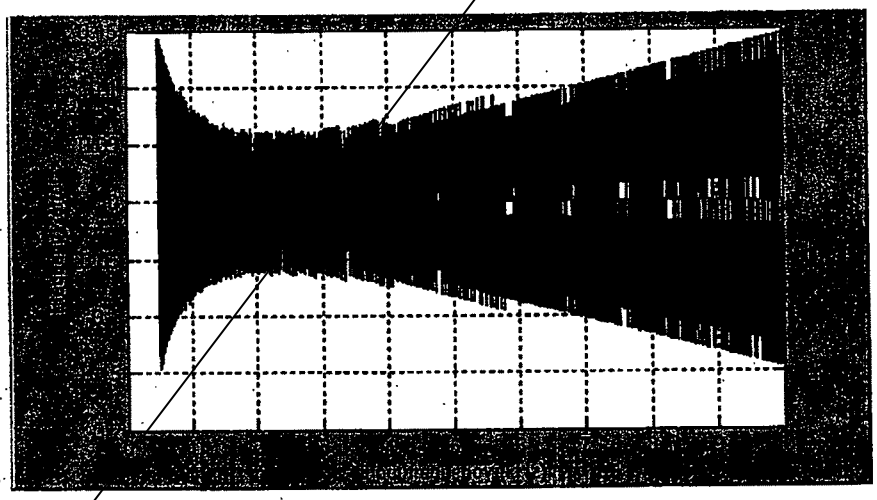


FIG. 326

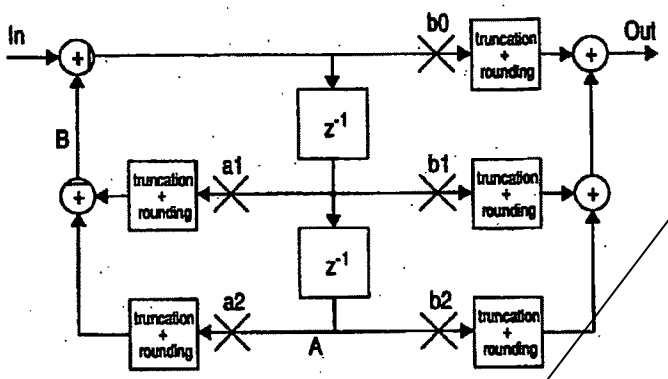


FIG. 327

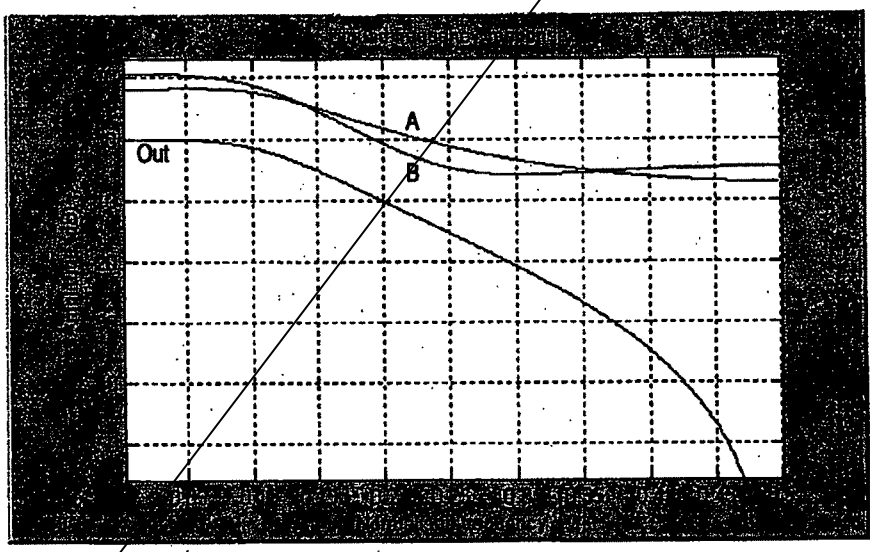


FIG. 328

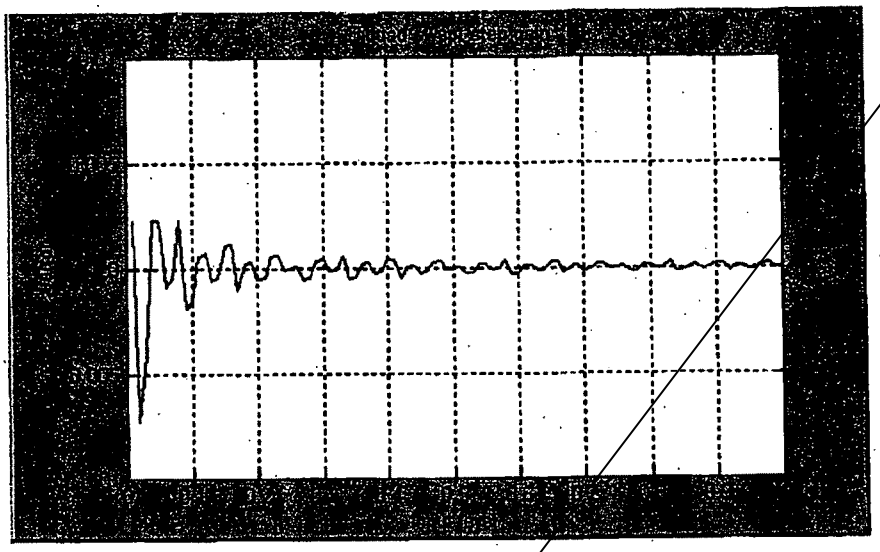


FIG. 329

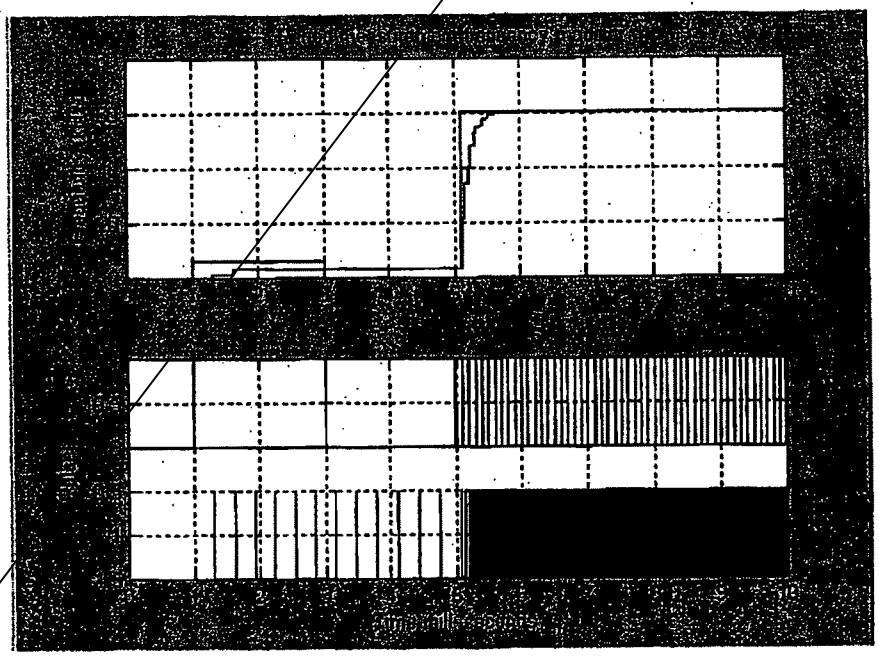


FIG. 330

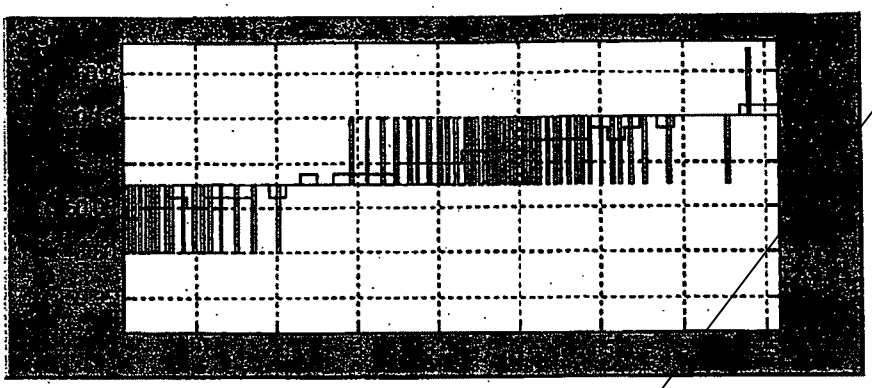


FIG. 331

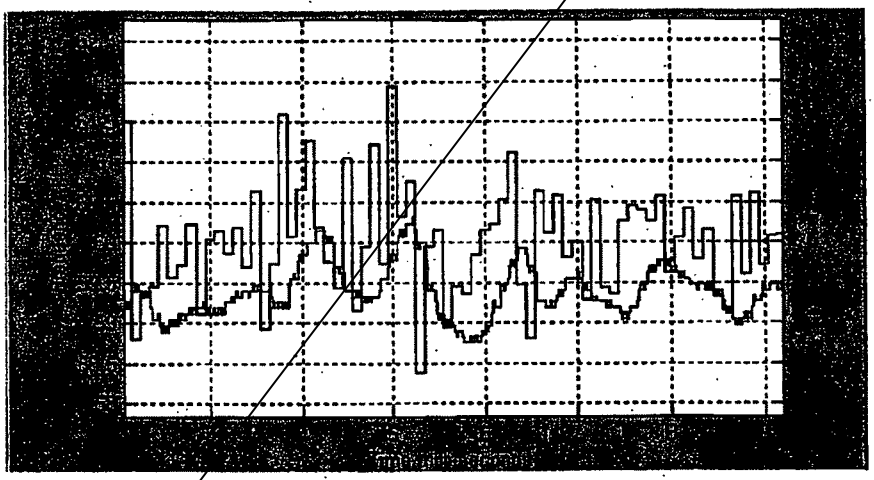


FIG. 332

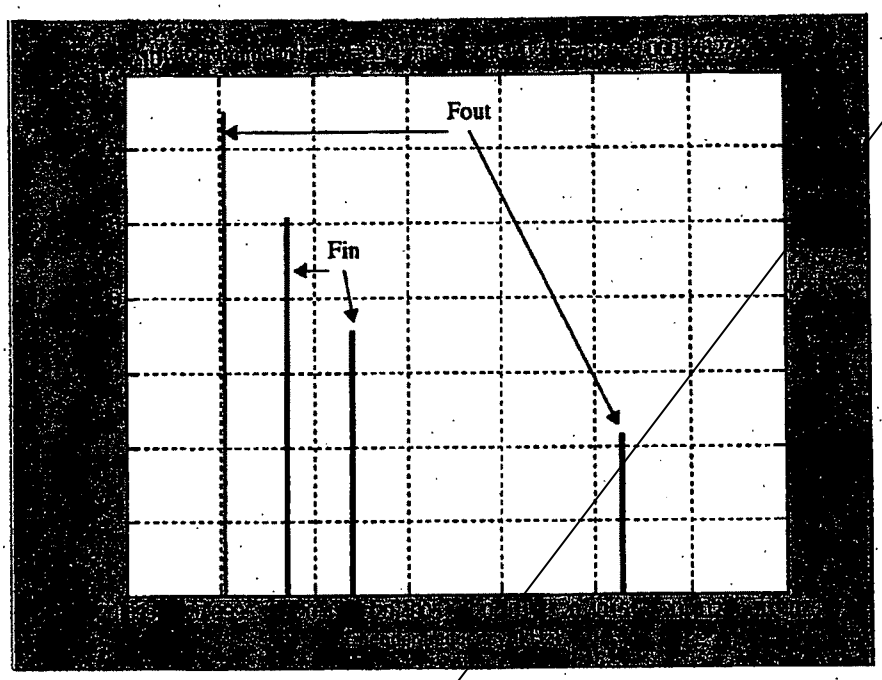


FIG. 333

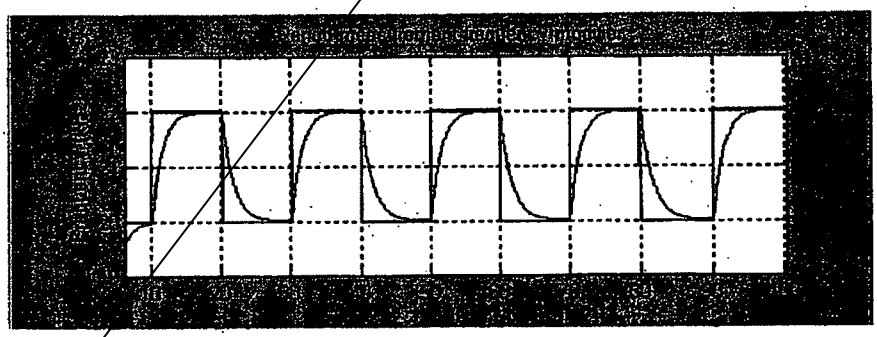


FIG. 334

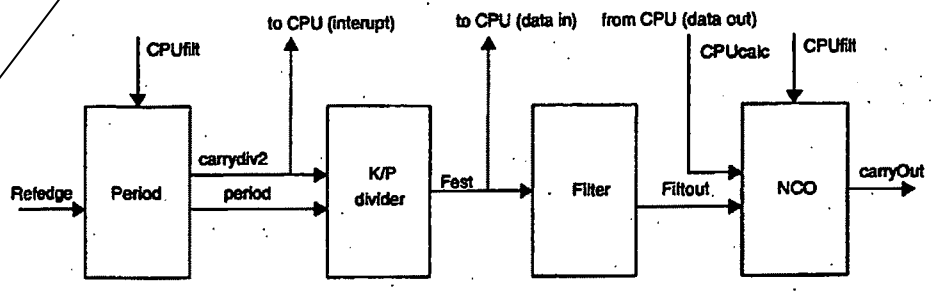


FIG. 335

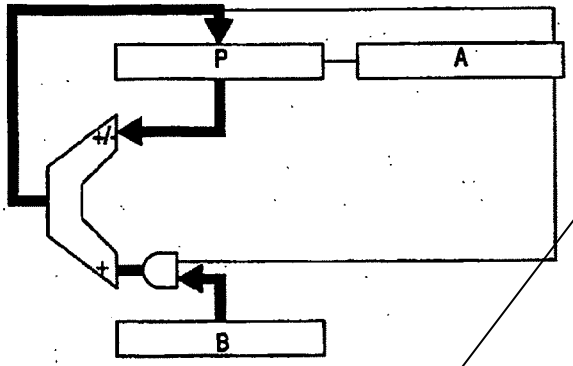


FIG. 336

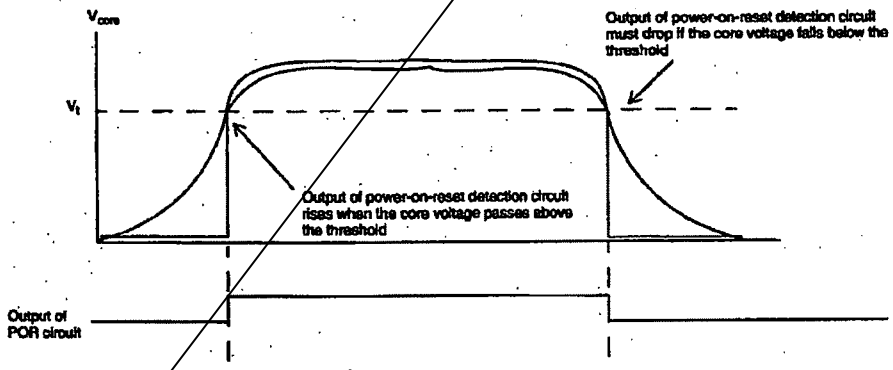


FIG. 337

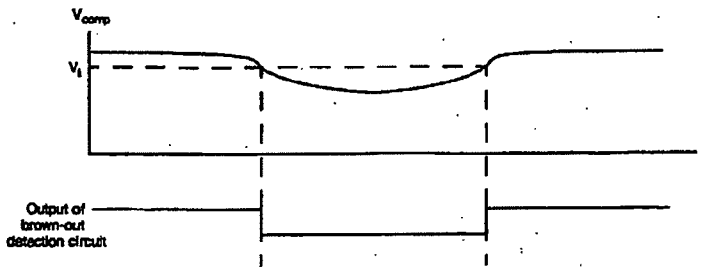
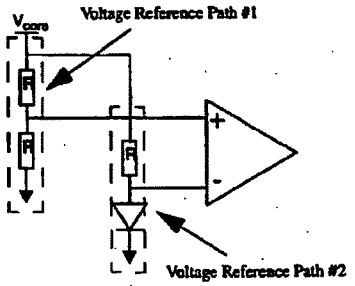
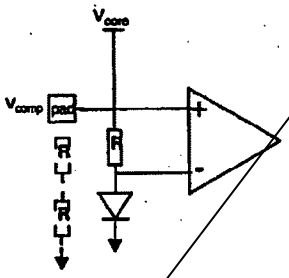


FIG. 338



Existing IBM POR macro



Modified IBM POR macro for brown-out detection, on external pin. [Voltage divider disconnected - minimal change].

FIG. 339

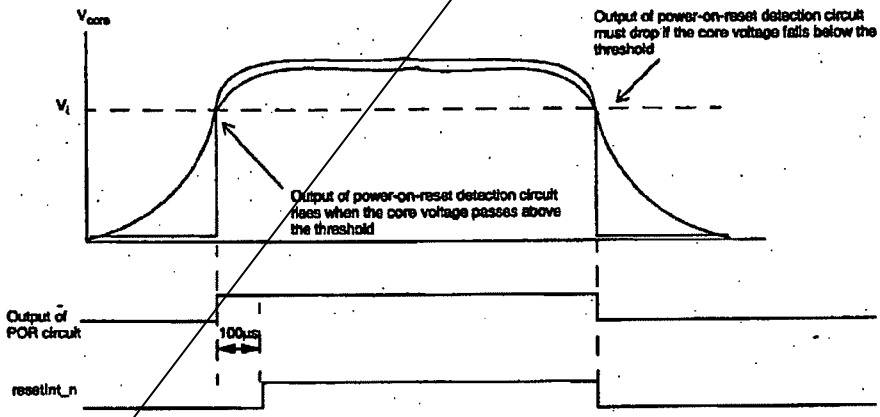


FIG. 340

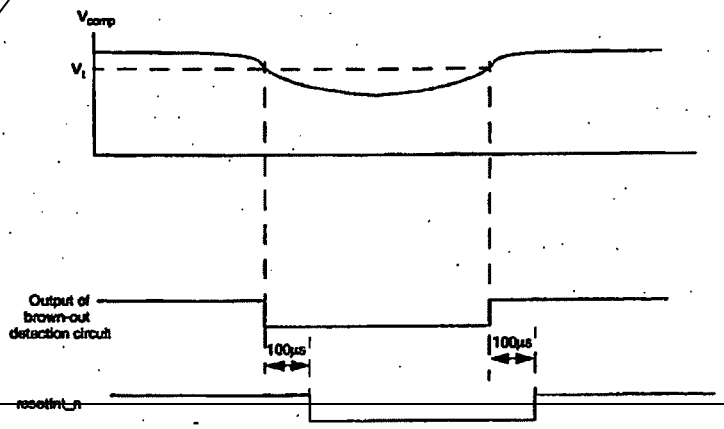


FIG. 341

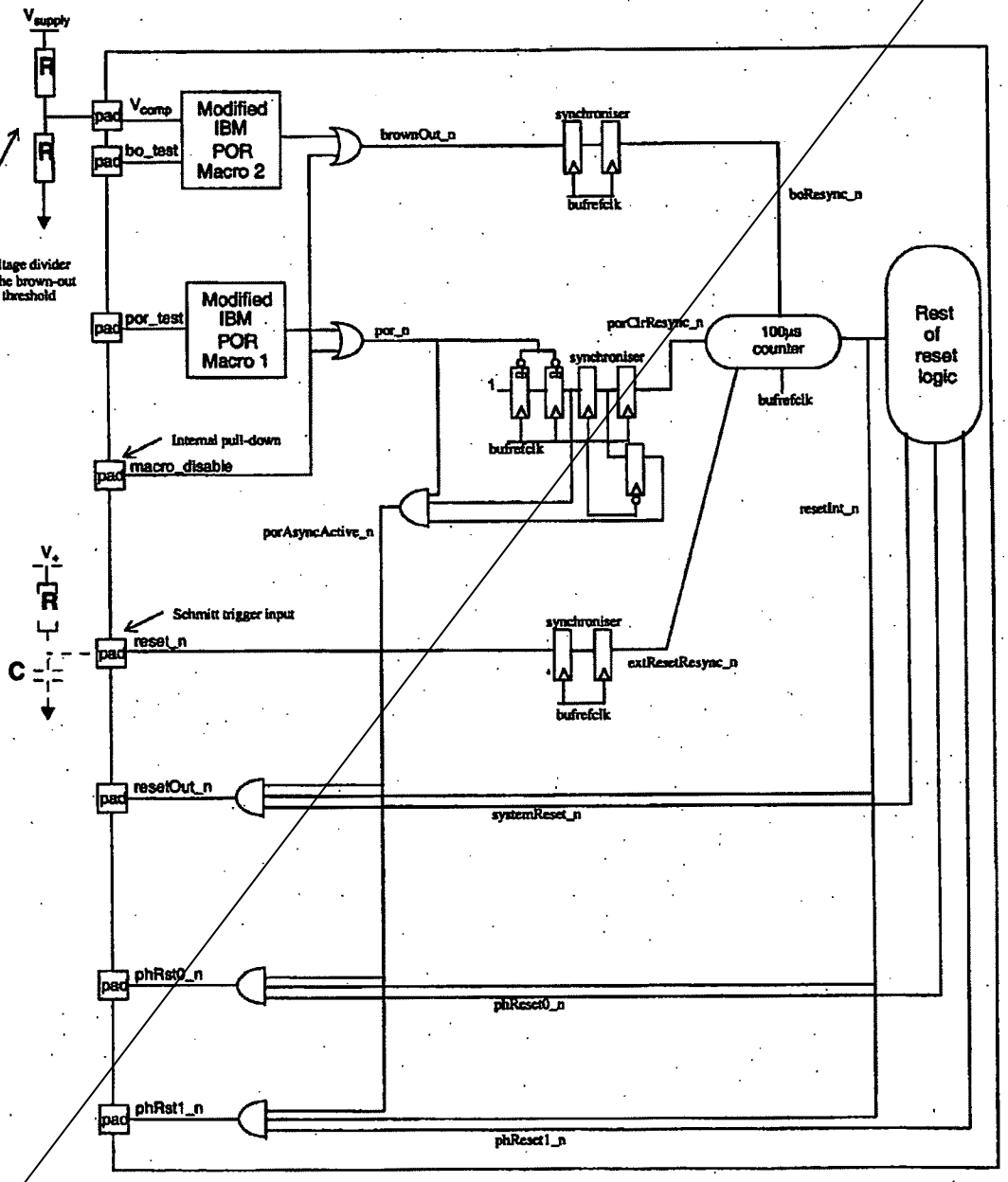

V supply
R
R
pad
pad
V comp
bo_test
Modified IBM POR Macro 2
brownOut_n
synchroniser
bufrefclk
boResync_n
External voltage divider determines the brown-out activation threshold
pad
por_test
Modified IBM POR Macro 1
por_n
porClrResync_n
100µs counter
Rest of reset logic
synchroniser
1
bufrefclk
bufrefclk
Internal pull-down
pad
macro_disable
porAsyncActive_n
resetInt_n
V+
R
C
Schmitt trigger input
pad
reset_n
synchroniser
extResetResync_n
bufrefclk
pad
resetOut_n
systemReset_n
pad
phRst0_n
phReset0_n
pad
phRst1_n
phReset1_n

FIG. 342

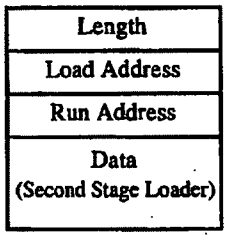


FIG. 343

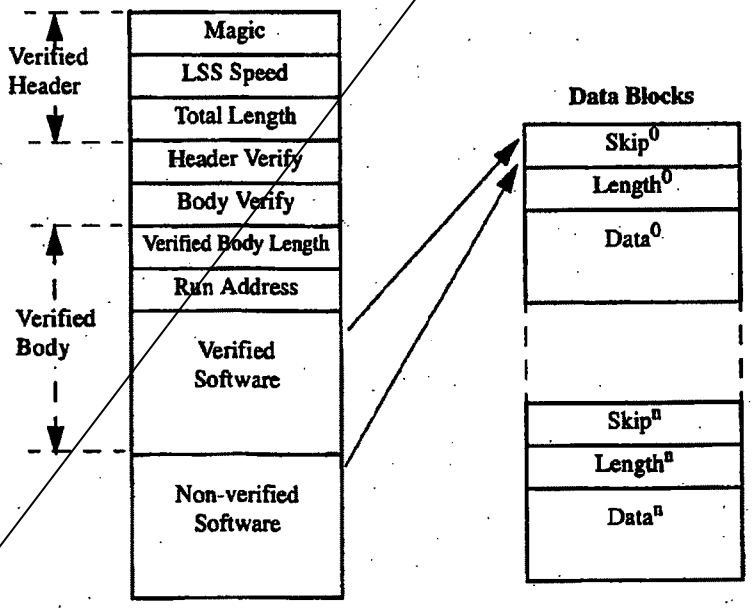


FIG. 344

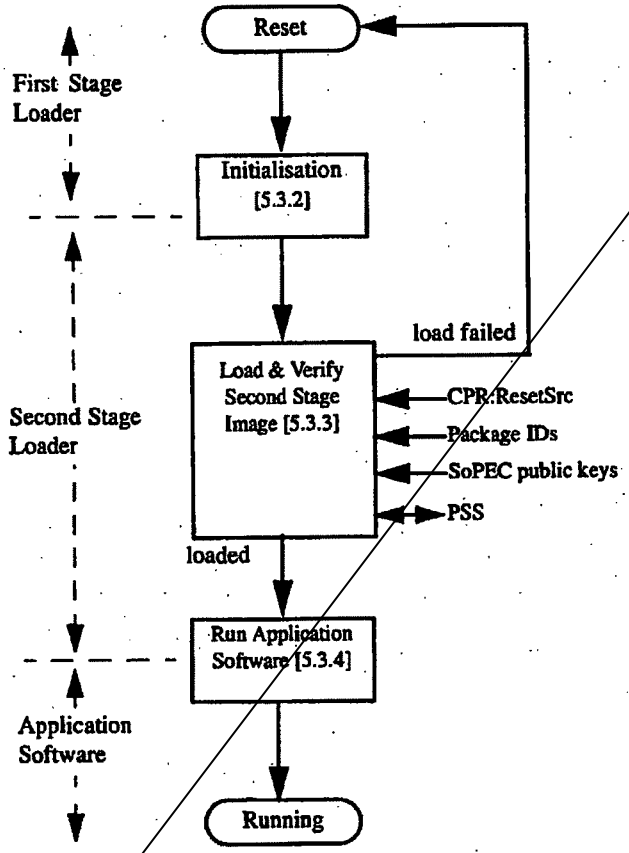

Reset
First Stage Loader
Initialisation [5.3.2]
load failed
Load & Verify Second Stage Image [5.3.3]
CPR.ResetSrc
Package IDs
SoPEC public keys
PSS
Second Stage Loader
loaded
Run Application Software [5.3.4]
Application Software
Running

FIG. 345

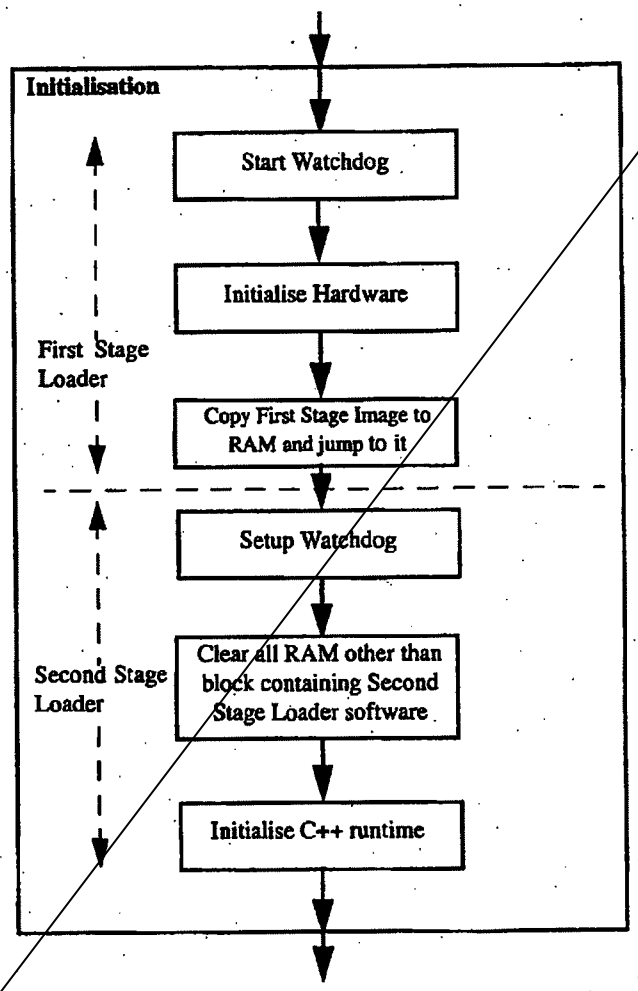

Initialisation
Start Watchdog
Initialise Hardware
First Stage Loader
Copy First Stage Image to RAM and jump to it
Setup Watchdog
Clear all RAM other than block containing Second Stage Loader software
Second Stage Loader
Initialise C++ runtime

FIG. 346

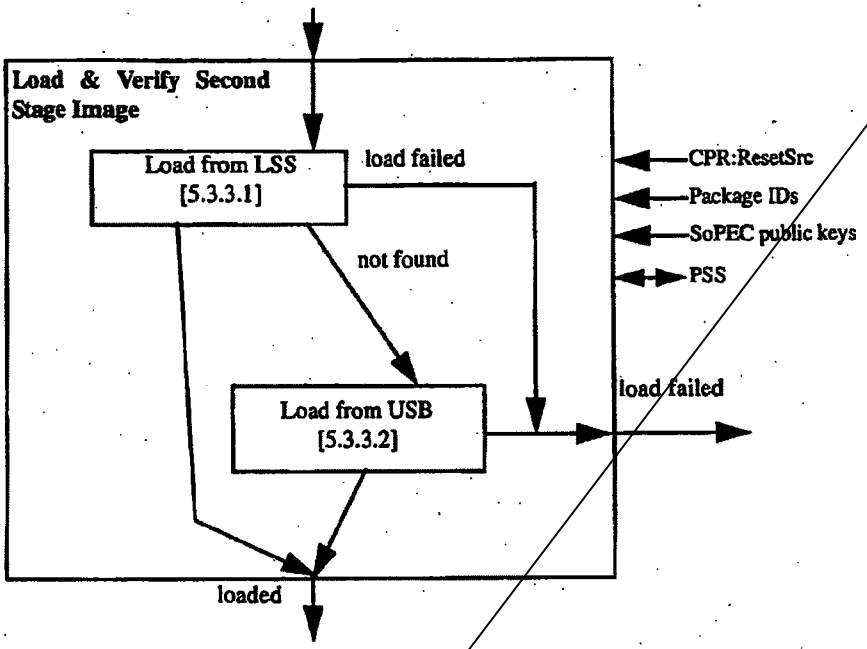


FIG. 347

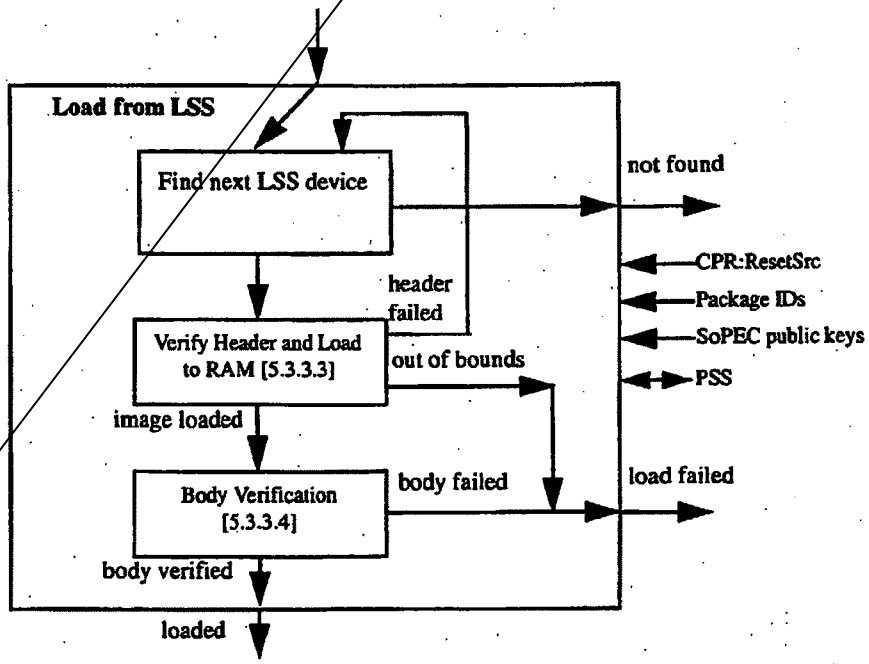


FIG. 348

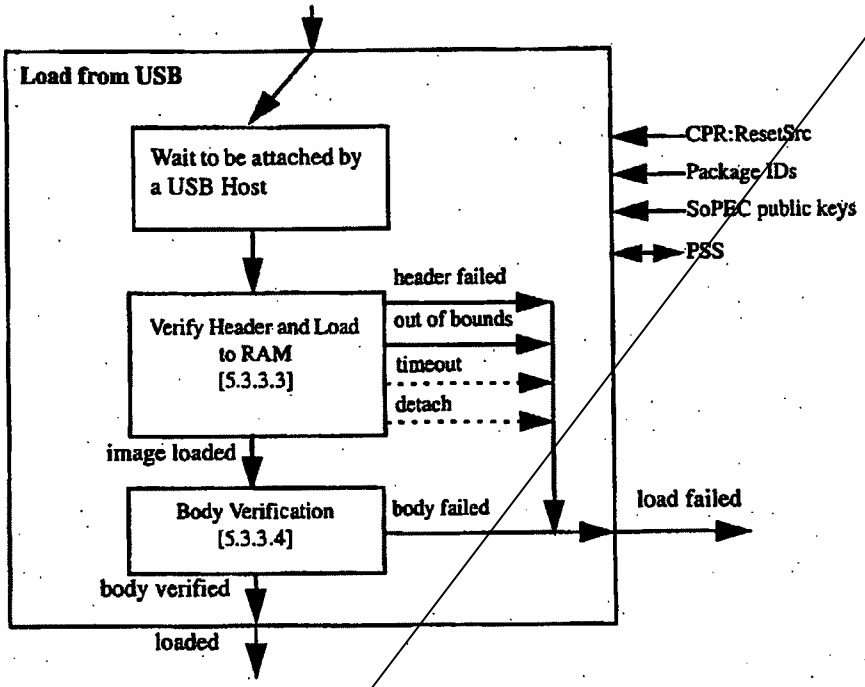

Load from USB
Wait to be attached by a USB Host
Verify Header and Load to RAM [5.3.3.3]
Body Verification [5.3.3.4]
header failed
out of bounds
timeout
detach
image loaded
body failed
load failed
body verified
loaded
CPR:ResetSrc
Package IDs
SoPEC public keys
PSS

FIG. 349

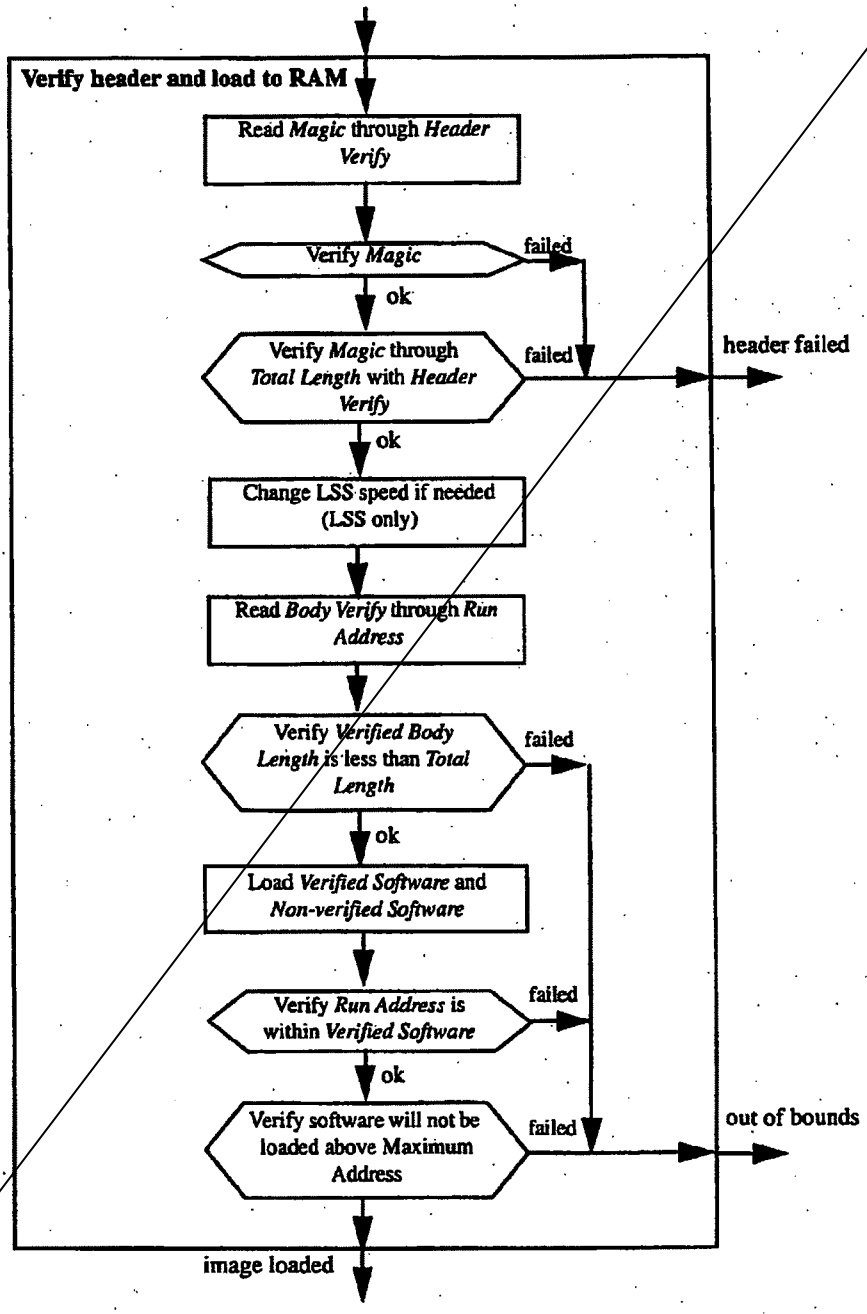

Verify header and load to RAM
Read Magic through Header Verify
Verify Magic
failed
ok
Verify Magic through Total Length with Header Verify
failed
header failed
ok
Change LSS speed if needed (LSS only)
Read Body Verify through Run Address
Verify Verified Body Length is less than Total Length
failed
ok
Load Verified Software and Non-verified Software
Verify Run Address is within Verified Software
failed
ok
Verify software will not be loaded above Maximum Address
failed
out of bounds
image loaded

FIG. 350

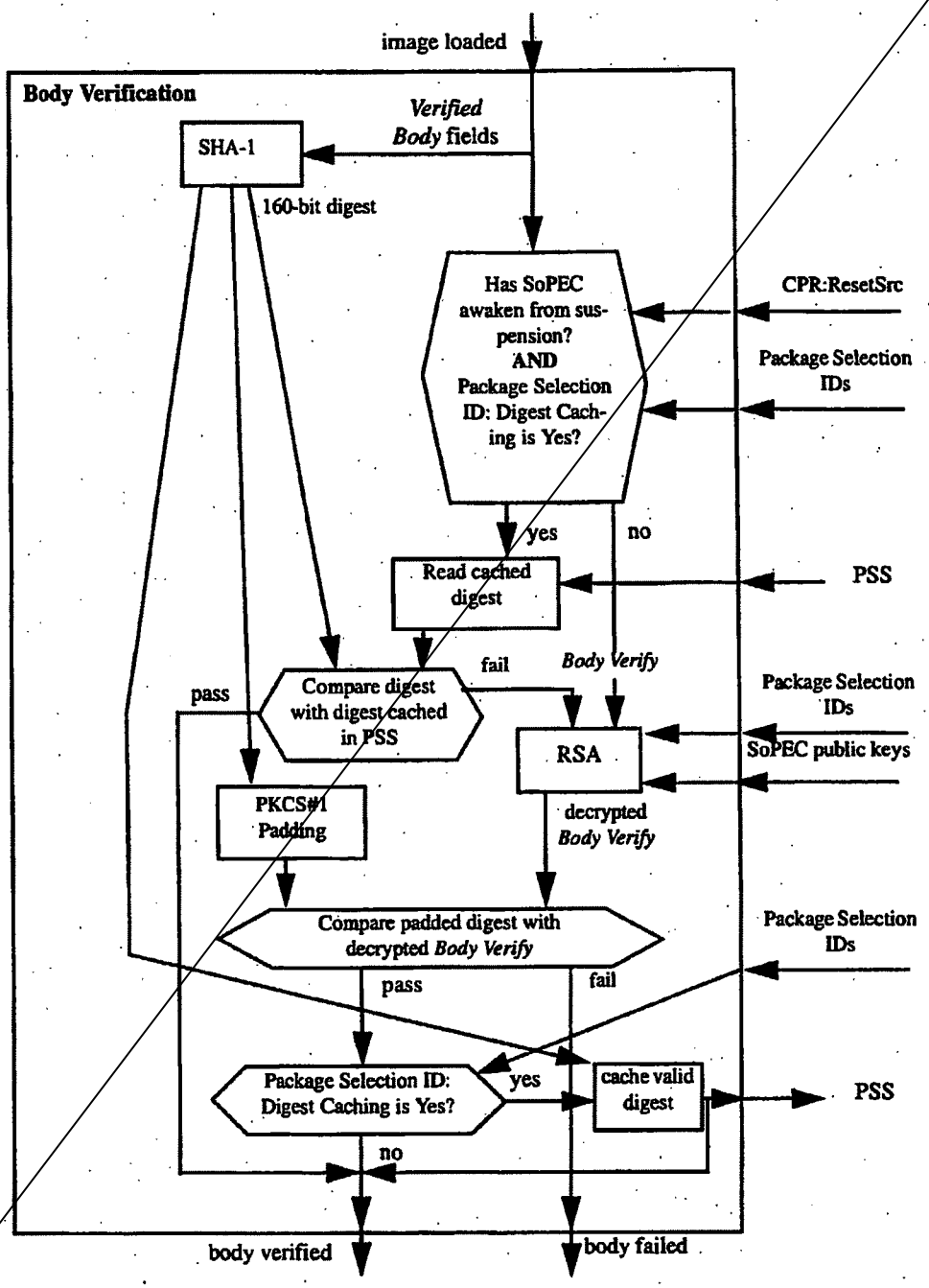

image loaded
Body Verification
Verified Body fields
SHA-1
160-bit digest
Has SoPEC awaken from suspension? AND Package Selection ID: Digest Caching is Yes?
CPR:ResetSrc
Package Selection IDs
yes
no
Read cached digest
PSS
fail
Body Verify
pass
Compare digest with digest cached in PSS
Package Selection IDs
RSA
SoPEC public keys
PKCS#1 Padding
decrypted Body Verify
Compare padded digest with decrypted Body Verify
Package Selection IDs
pass
fail
Package Selection ID: Digest Caching is Yes?
yes
cache valid digest
PSS
no
body verified
body failed

FIG. 351

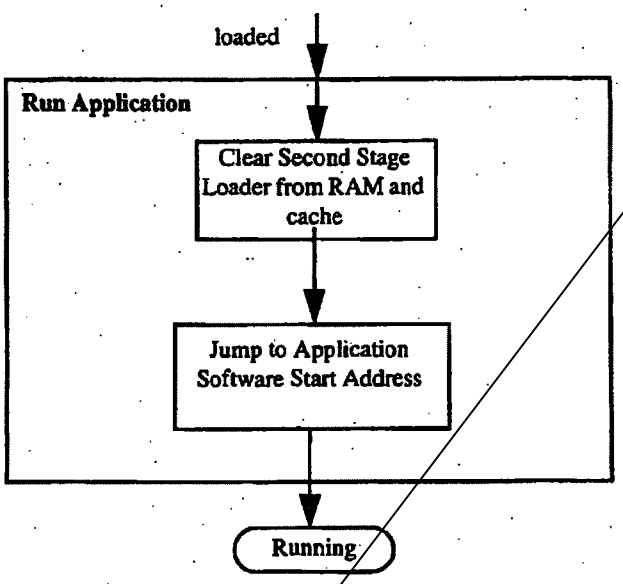


FIG. 352

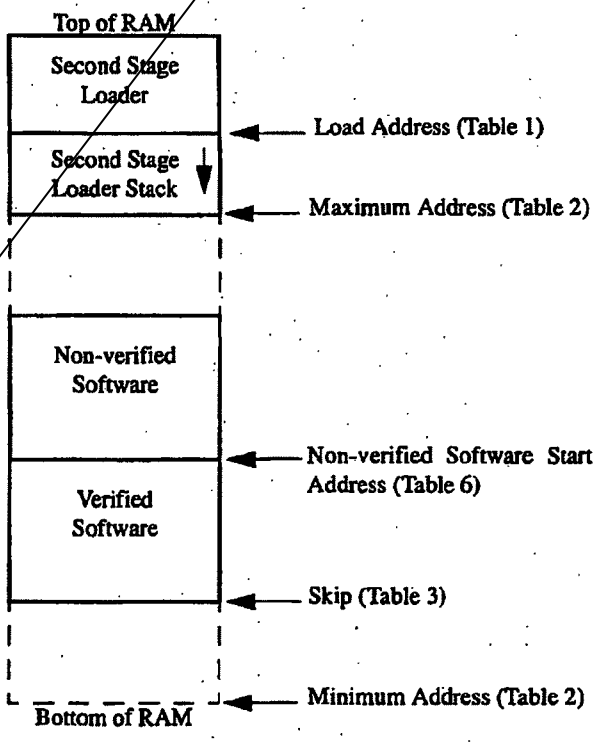


FIG. 353

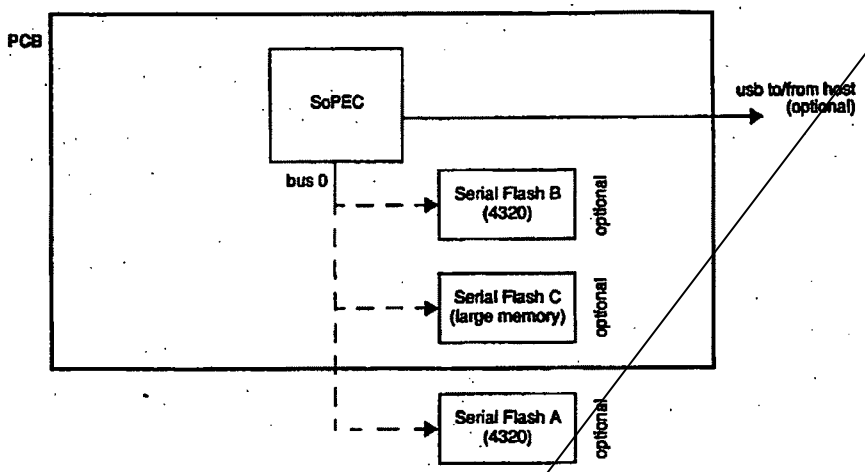

PCB
SoPEC
usb to/from host
(optional)
bus 0
Serial Flash B
(4320)
optional
Serial Flash C
(large memory)
optional
Serial Flash A
(4320)
optional

FIG. 354

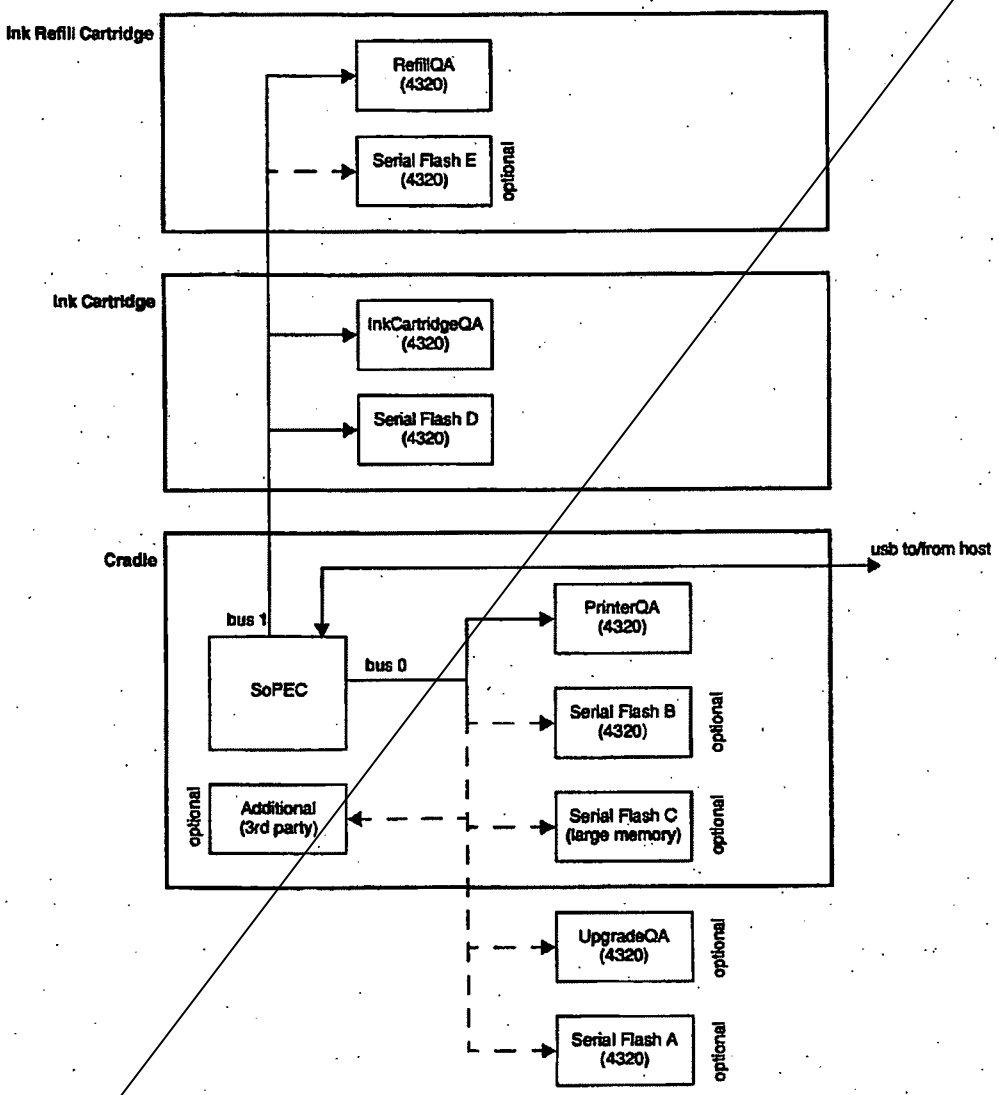

Ink Refill Cartridge
RefillQA (4320)
Serial Flash E (4320)
optional
Ink Cartridge
InkCartridgeQA (4320)
Serial Flash D (4320)
Cradle
usb to/from host
bus 1
SoPEC
bus 0
PrinterQA (4320)
Serial Flash B (4320)
optional
optional
Additional (3rd party)
Serial Flash C (large memory)
optional
UpgradeQA (4320)
optional
Serial Flash A (4320)
optional

FIG. 355

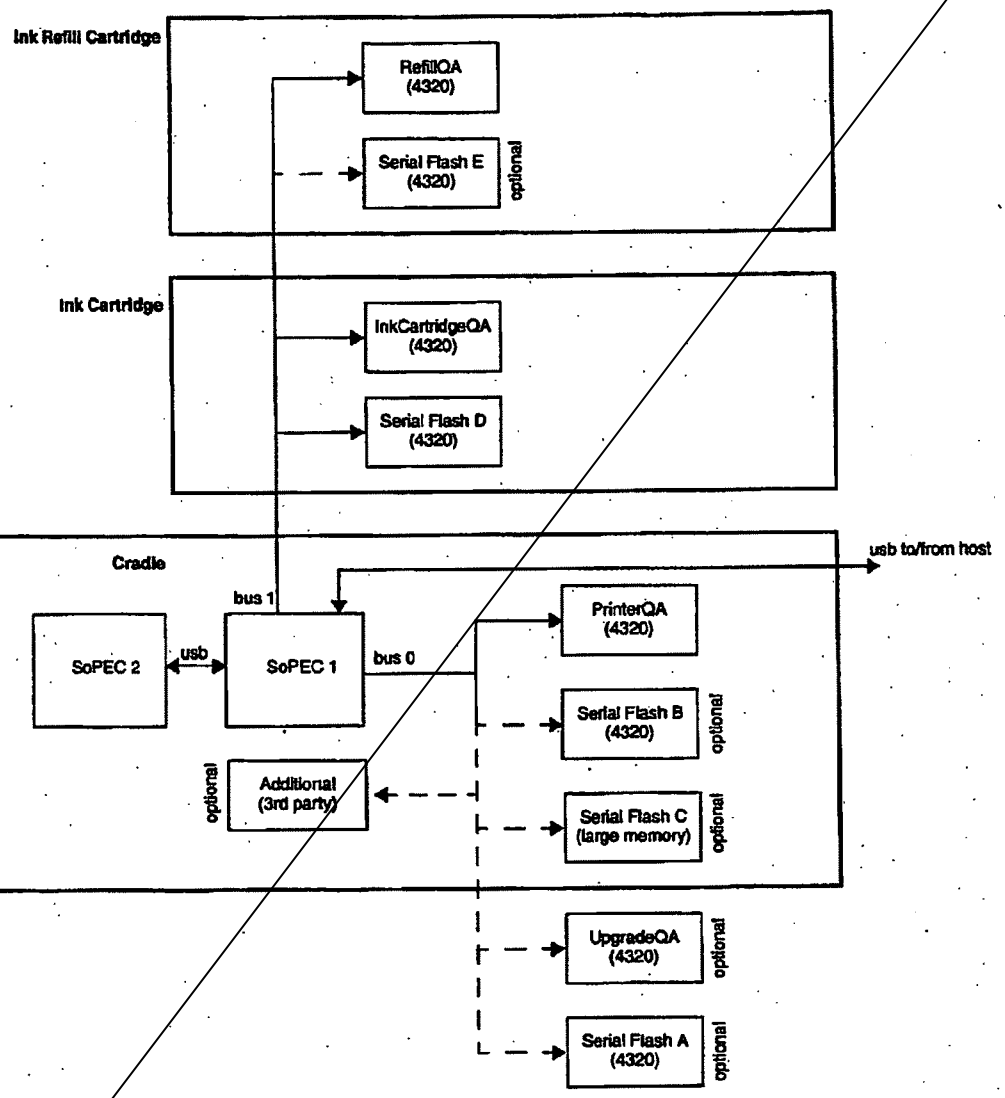

Ink Refill Cartridge
RefillQA (4320)
Serial Flash E (4320)
optional
Ink Cartridge
InkCartridgeQA (4320)
Serial Flash D (4320)
Cradle
usb to/from host
bus 1
SoPEC 2
usb
SoPEC 1
bus 0
PrinterQA (4320)
Serial Flash B (4320)
optional
optional
Additional (3rd party)
Serial Flash C (large memory)
optional
UpgradeQA (4320)
optional
Serial Flash A (4320)
optional

FIG. 356

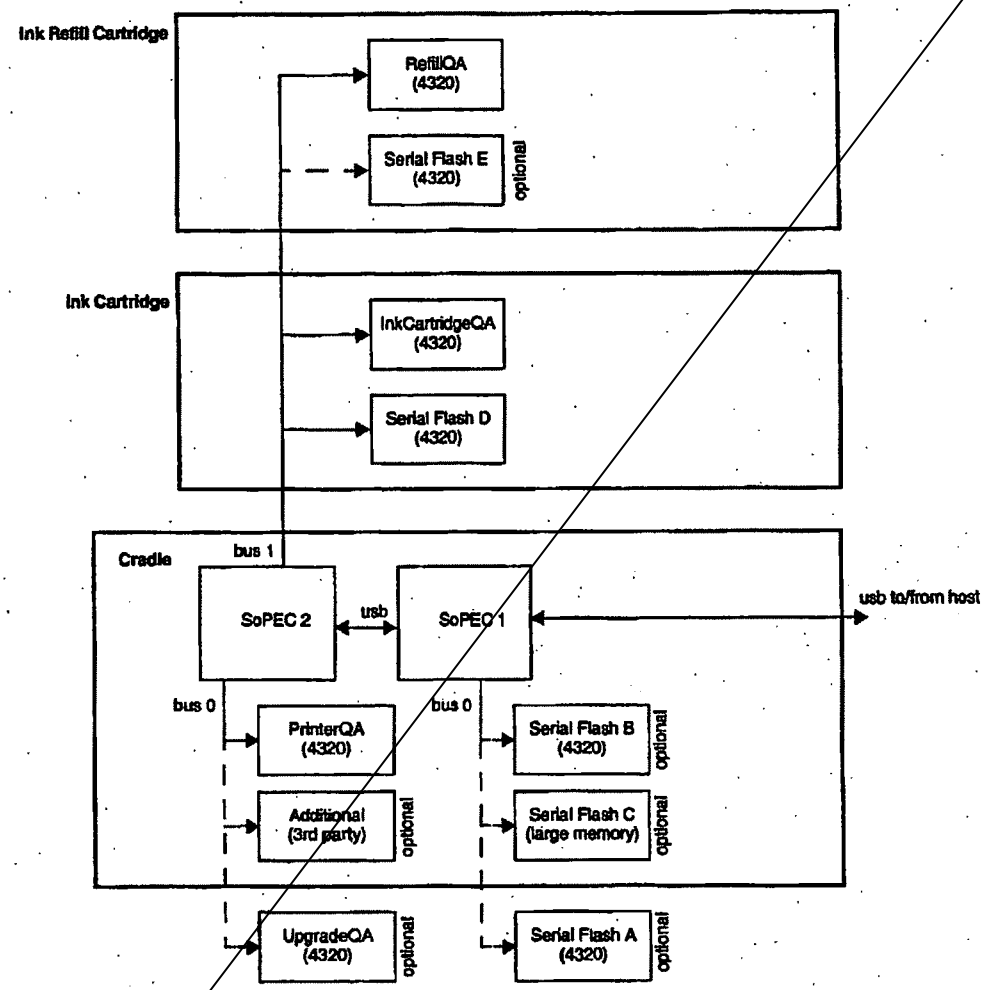

Ink Refill Cartridge
RefillQA (4320)
Serial Flash E (4320)
optional
Ink Cartridge
InkCartridgeQA (4320)
Serial Flash D (4320)
Cradle
bus 1
SoPEC 2
usb
SoPEC 1
usb to/from host
bus 0
PrinterQA (4320)
bus 0
Serial Flash B (4320)
optional
Additional (3rd party)
optional
Serial Flash C (large memory)
optional
UpgradeQA (4320)
optional
Serial Flash A (4320)
optional

FIG. 357

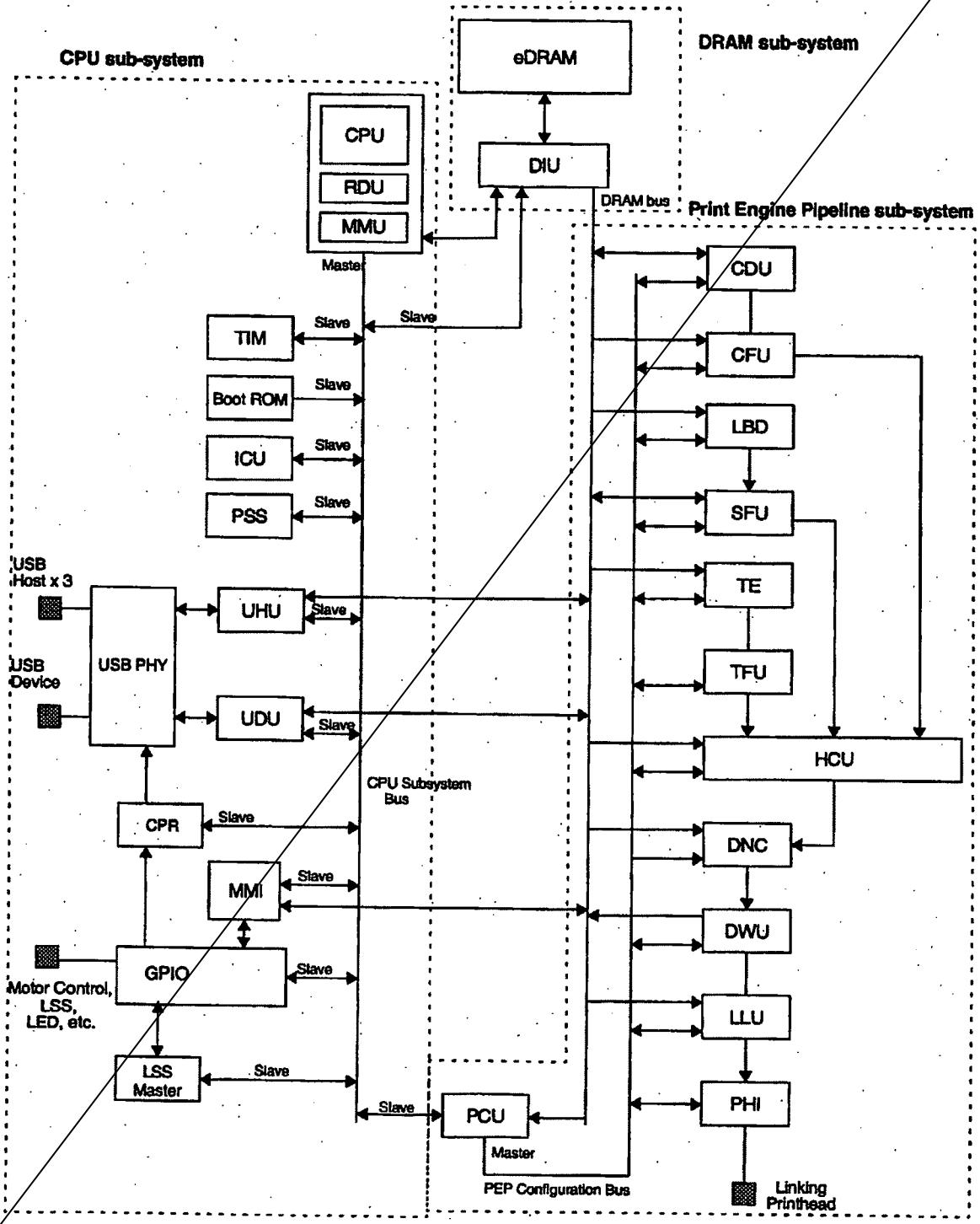

CPU sub-system
DRAM sub-system
Print Engine Pipeline sub-system
eDRAM
CPU
RDU
MMU
DIU
DRAM bus
Master
Slave
TIM
Boot ROM
ICU
PSS
USB Host x 3
USB Device
USB PHY
UHU
UDU
CPU Subsystem Bus
CPR
MMI
GPIO
Motor Control, LSS, LED, etc.
LSS Master
PCU
Master
PEP Configuration Bus
CDU
CFU
LBD
SFU
TE
TFU
HCU
DNC
DWU
LLU
PHI
Linking Printhead

FIG. 358

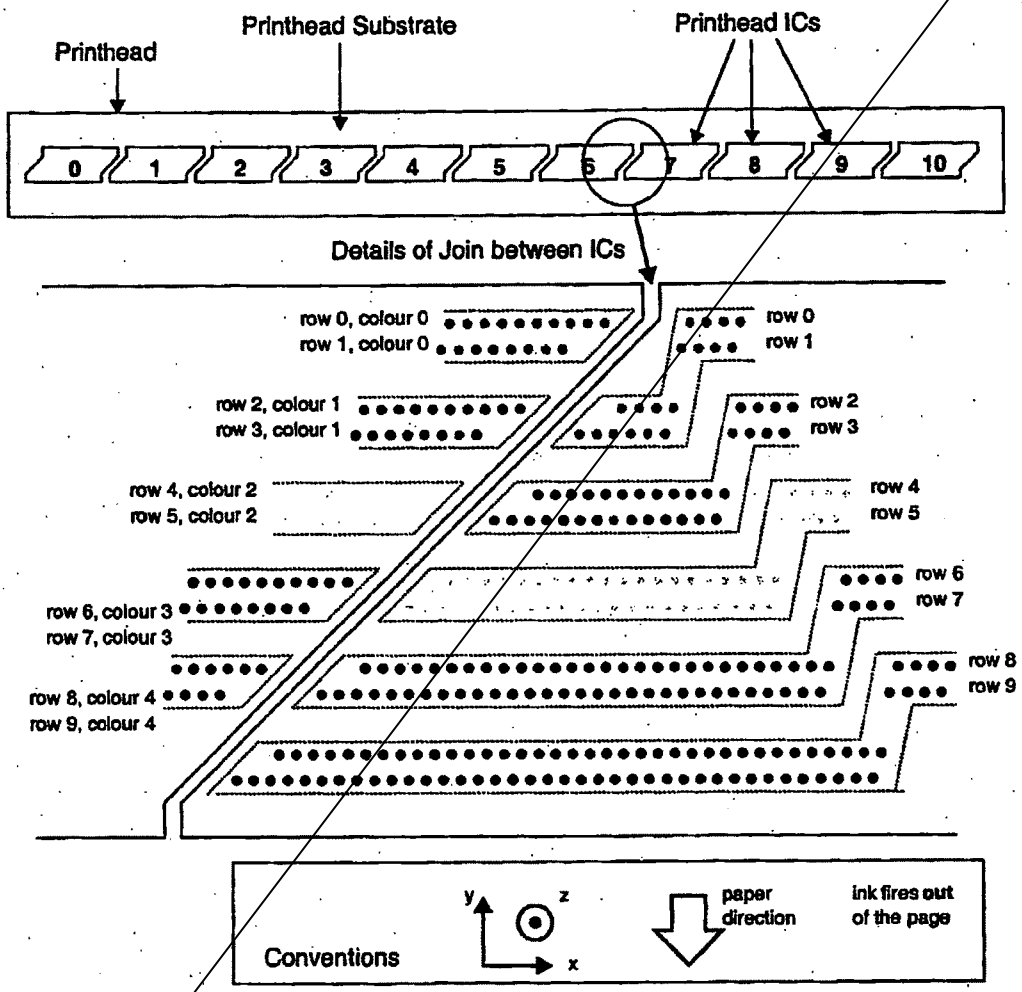

Printhead
Printhead Substrate
Printhead ICs
0
1
2
3
4
5
6
7
8
9
10
Details of Join between ICs
row 0, colour 0
row 1, colour 0
row 0
row 1
row 2, colour 1
row 3, colour 1
row 2
row 3
row 4, colour 2
row 5, colour 2
row 4
row 5
row 6, colour 3
row 7, colour 3
row 6
row 7
row 8, colour 4
row 9, colour 4
row 8
row 9
Conventions
y
z
x
paper direction
ink fires out of the page

FIG. 359

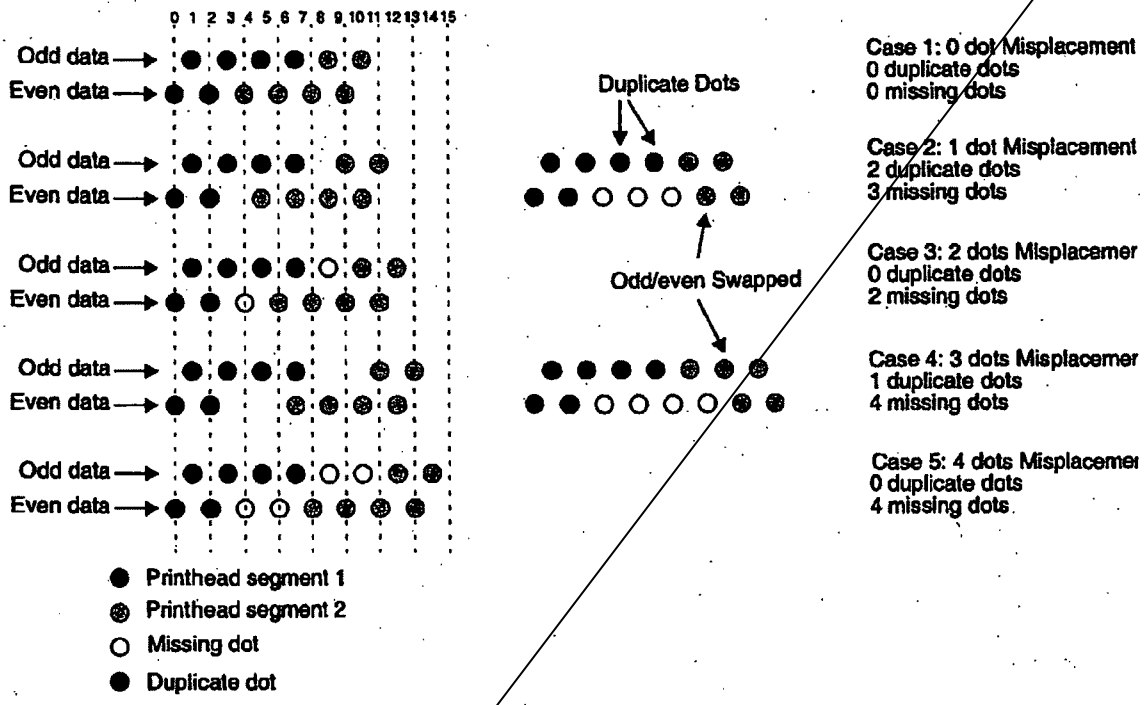


FIG. 360

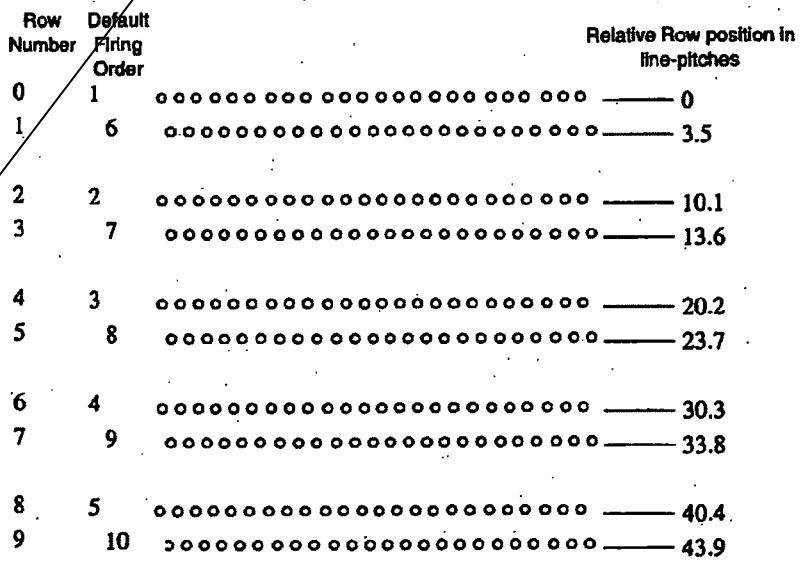


FIG. 361

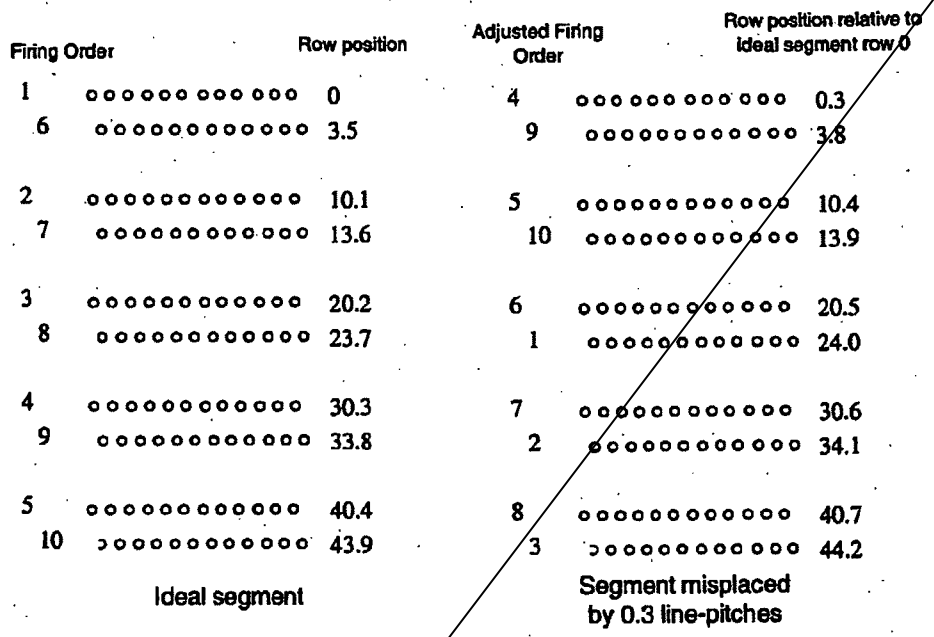


FIG. 362

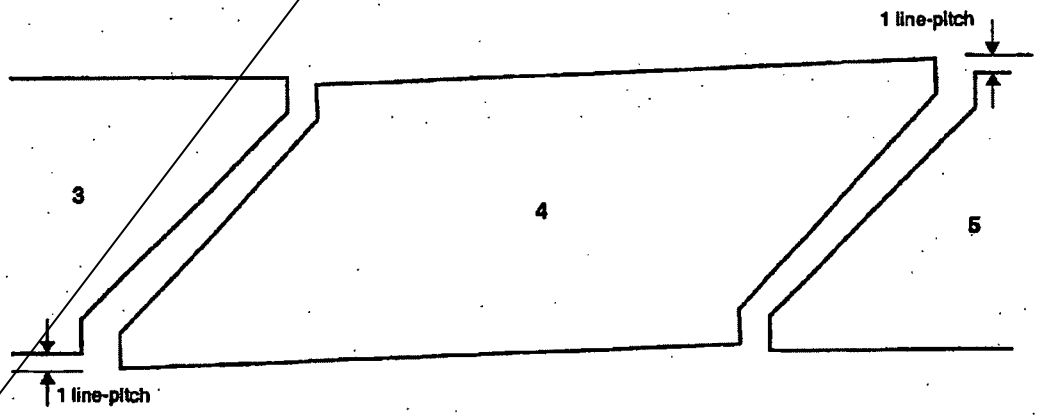


FIG. 363

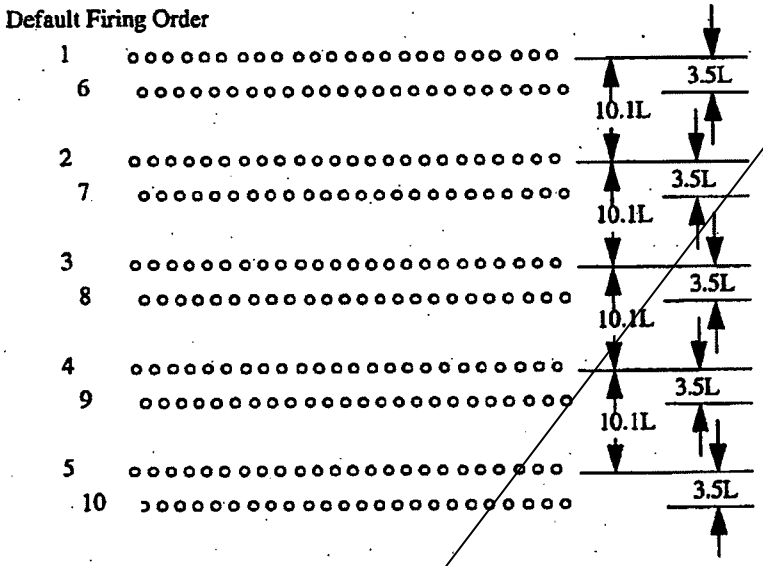


FIG. 364

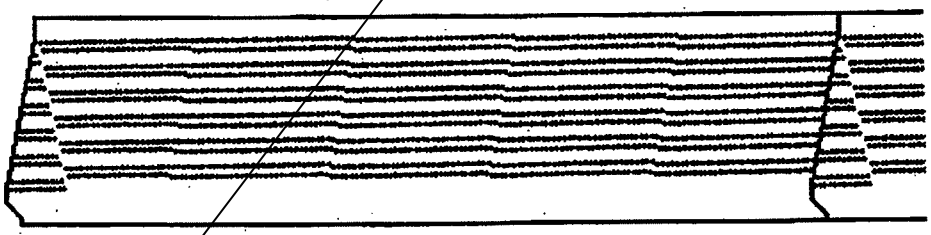


FIG. 365

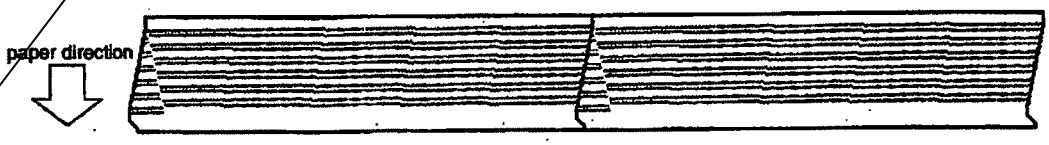


FIG. 366

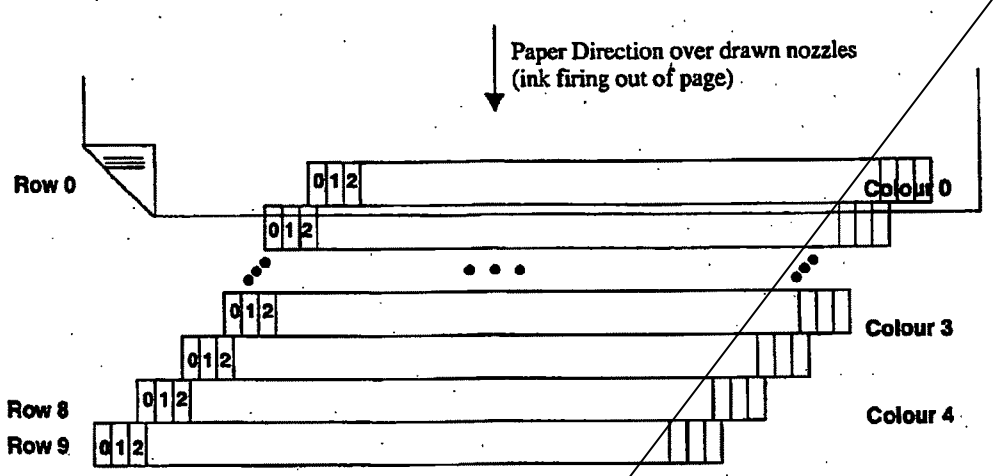


FIG. 367

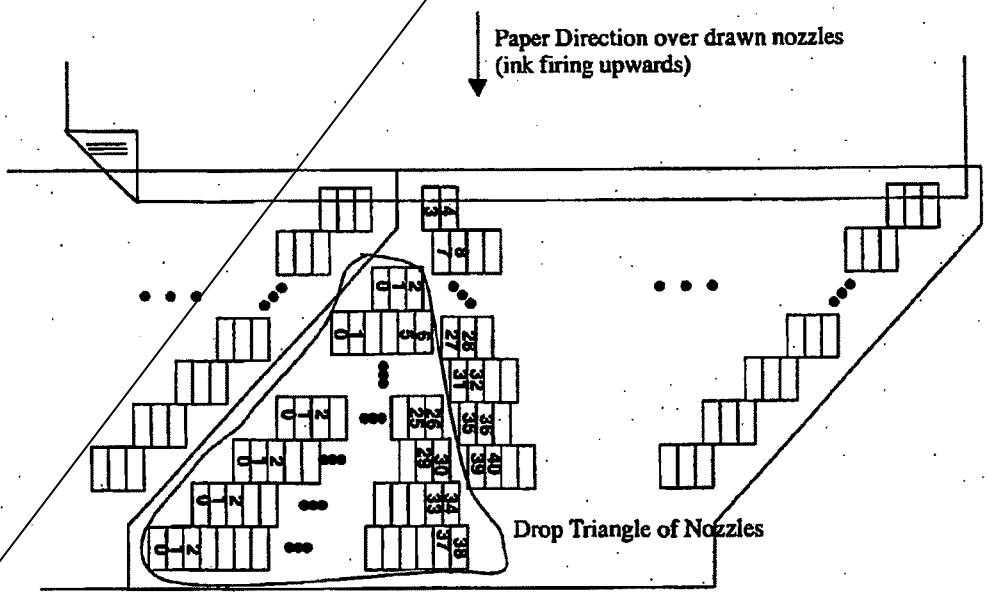


FIG. 368

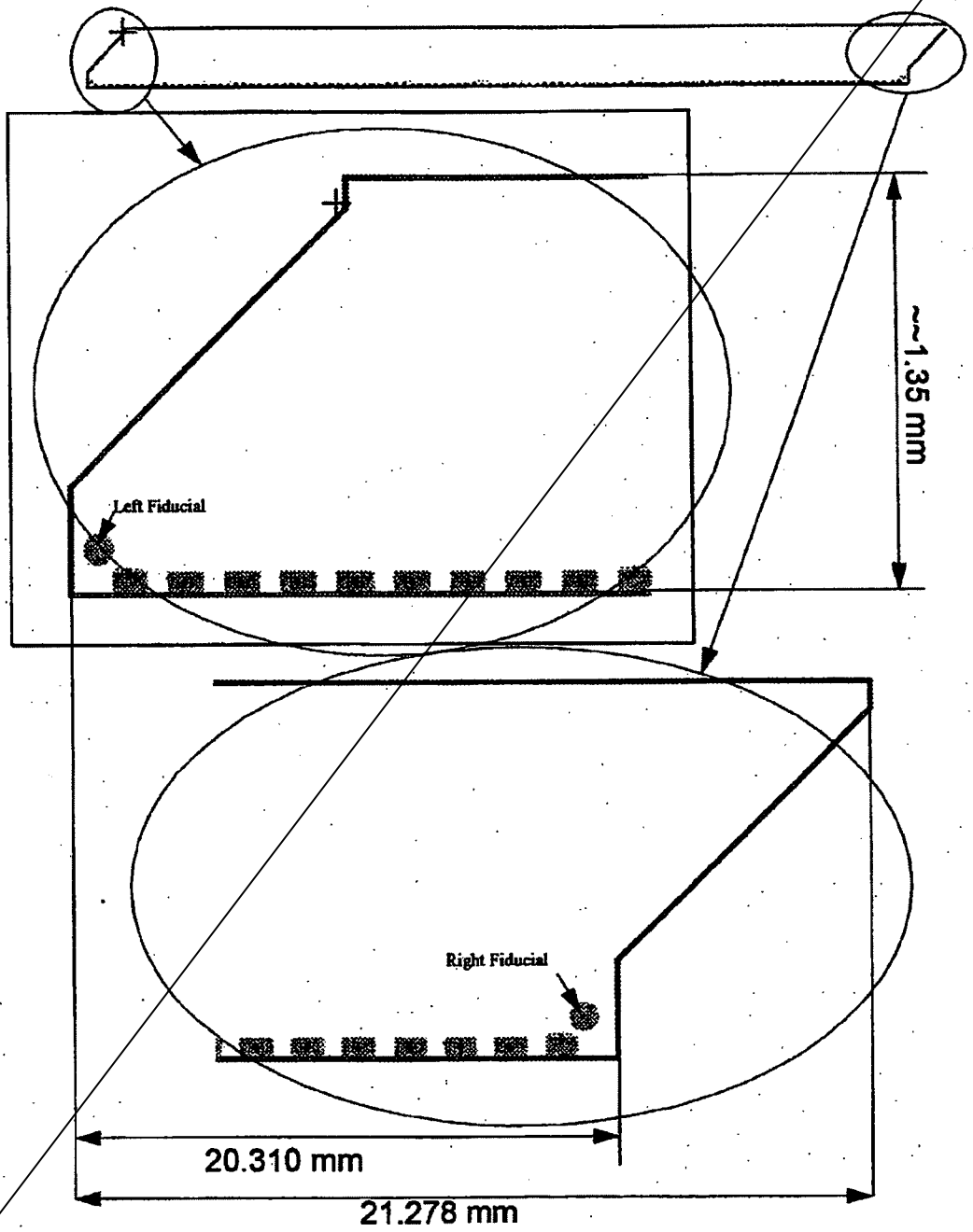


FIG. 369

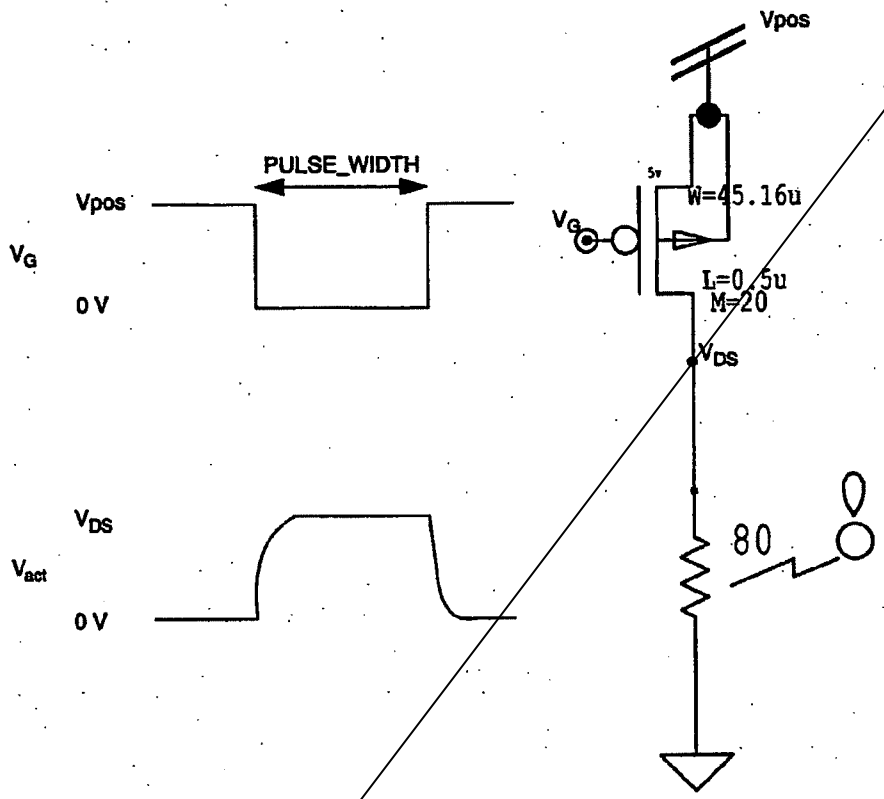


FIG. 370

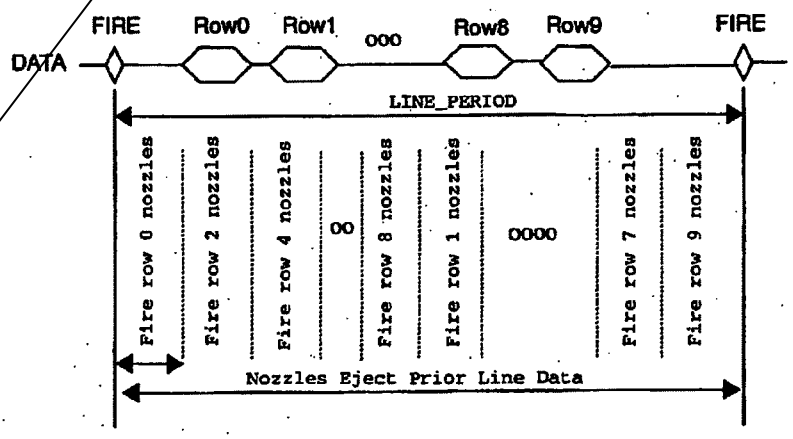


FIG. 371

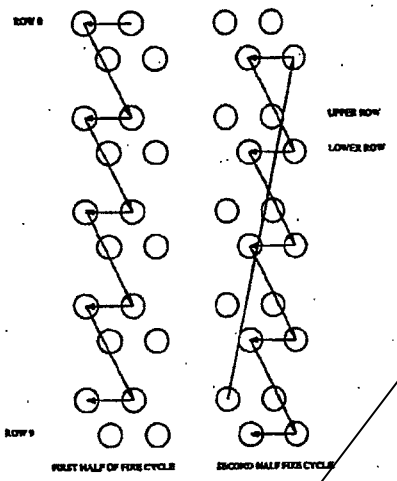


FIG. 372

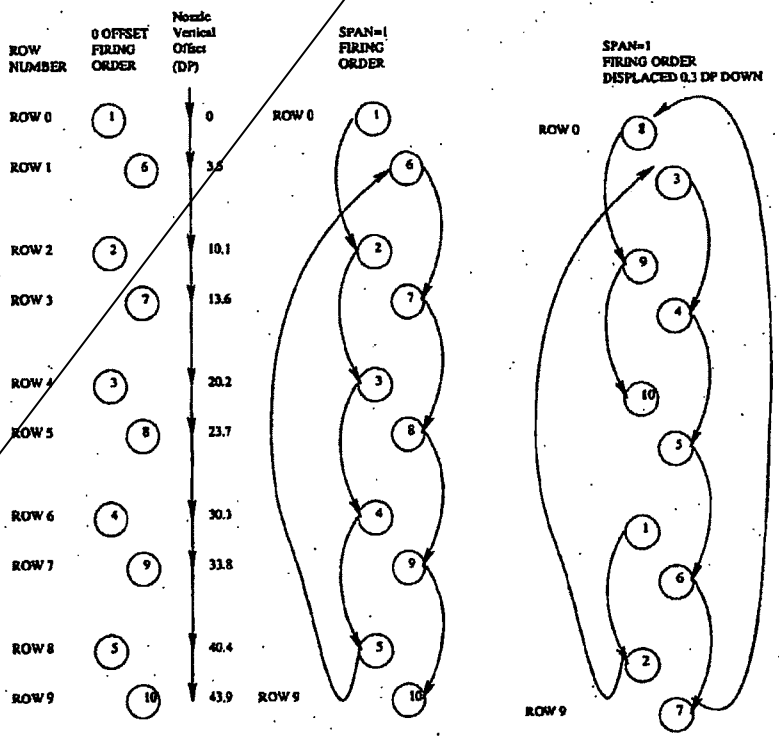


FIG. 373

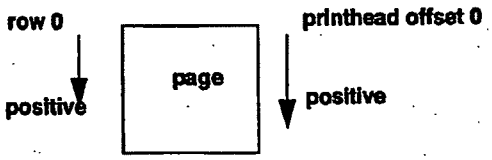


FIG. 374

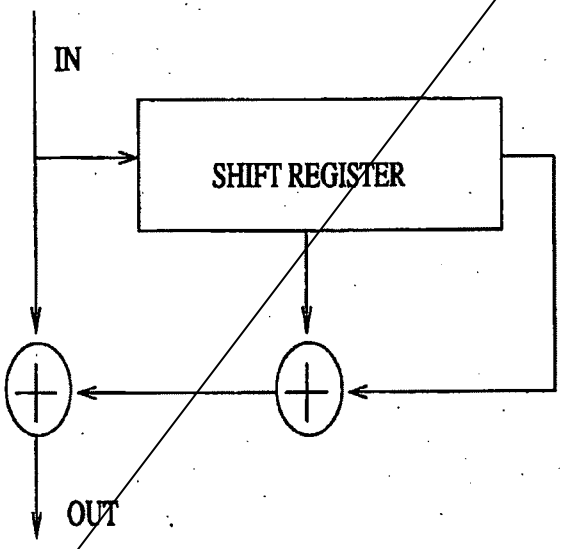


FIG. 375

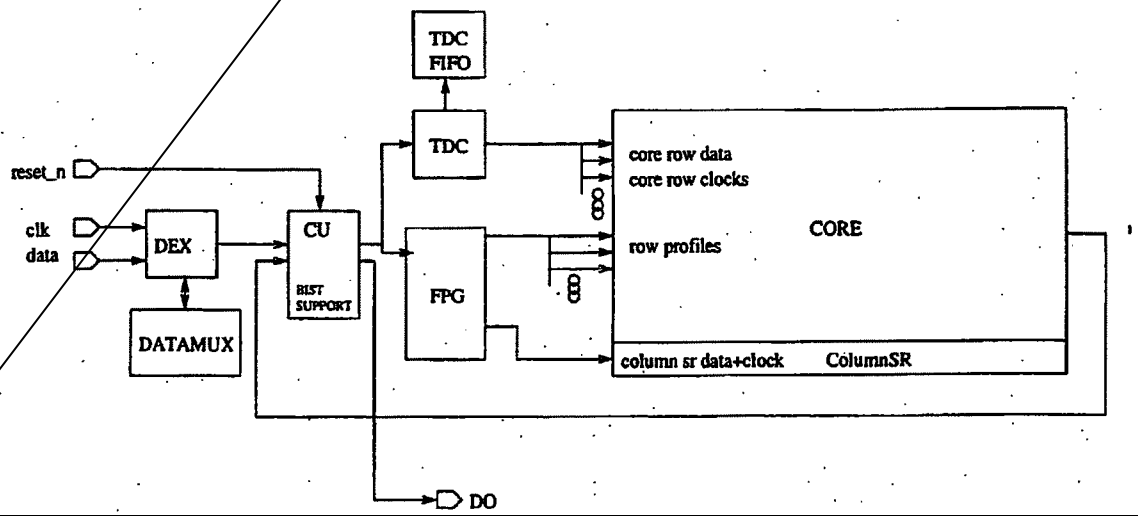


FIG. 376

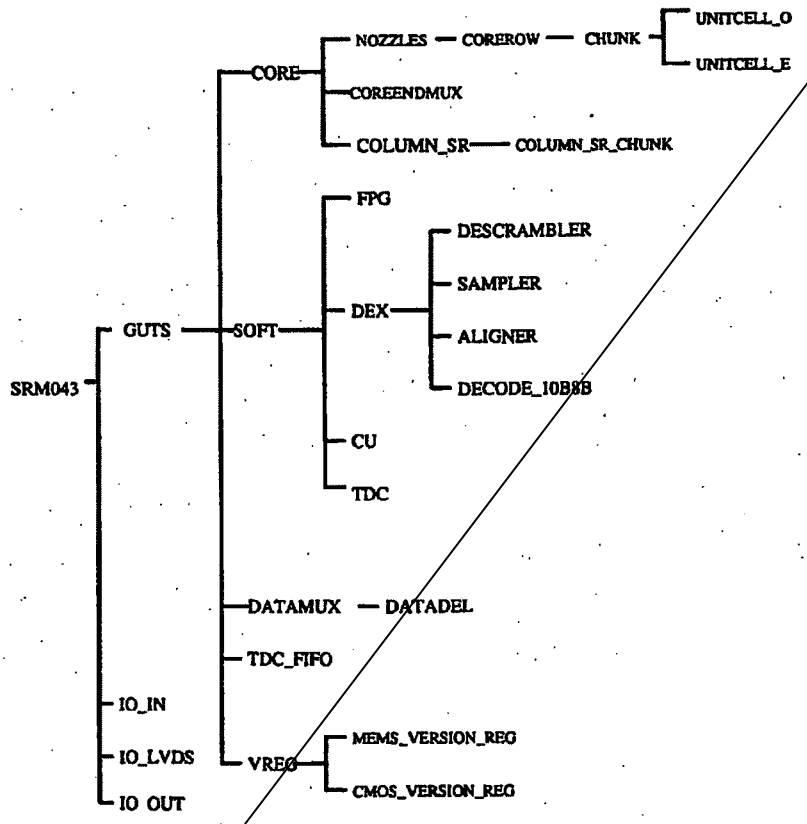

SRM043
GUTS
IO_IN
IO_LVDS
IO OUT
SOFT
CORE
NOZZLES
COREROW
CHUNK
UNITCELL_O
UNITCELL_E
COREENDMUX
COLUMN_SR
COLUMN_SR_CHUNK
FPG
DEX
DESCRAMBLER
SAMPLER
ALIGNER
DECODE_10B8B
CU
TDC
DATAMUX
DATADEL
TDC_FIFO
VREG
MEMS_VERSION_REG
CMOS_VERSION_REG

FIG. 377

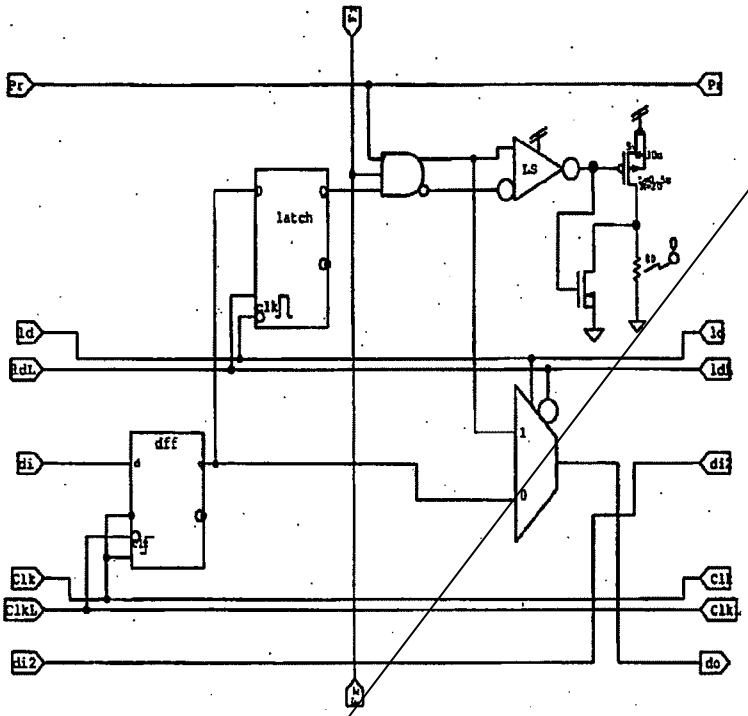


FIG. 378

	uu7		uu6		uu5		uu4		uu3		uu2	Pr	uu1	Ck	uu0	Ld
ul7	ul6		ul5		ul4		ul3		ul2		ul1	Pr	ul0	Ck		

FIG. 379

	FIRE
D0	D1
D1	D2
LD	LD
PR	PR
CK	CK
FIRE	

FIG. 380

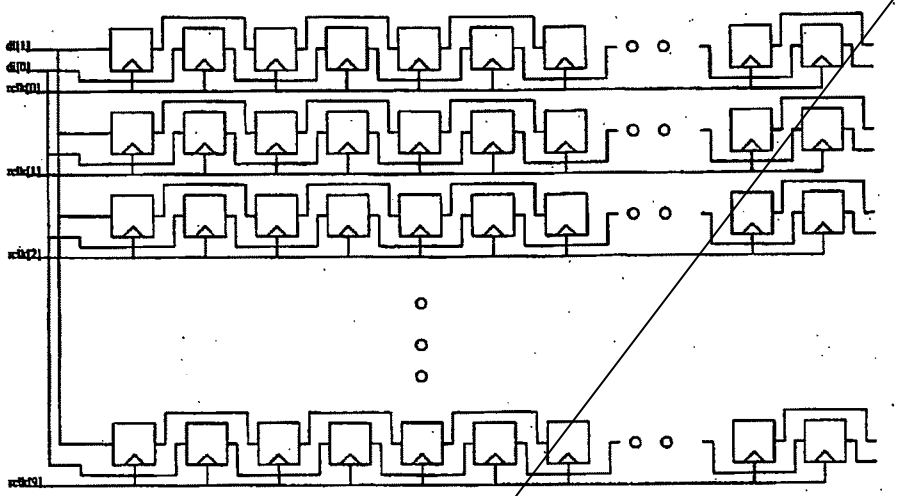


FIG. 381

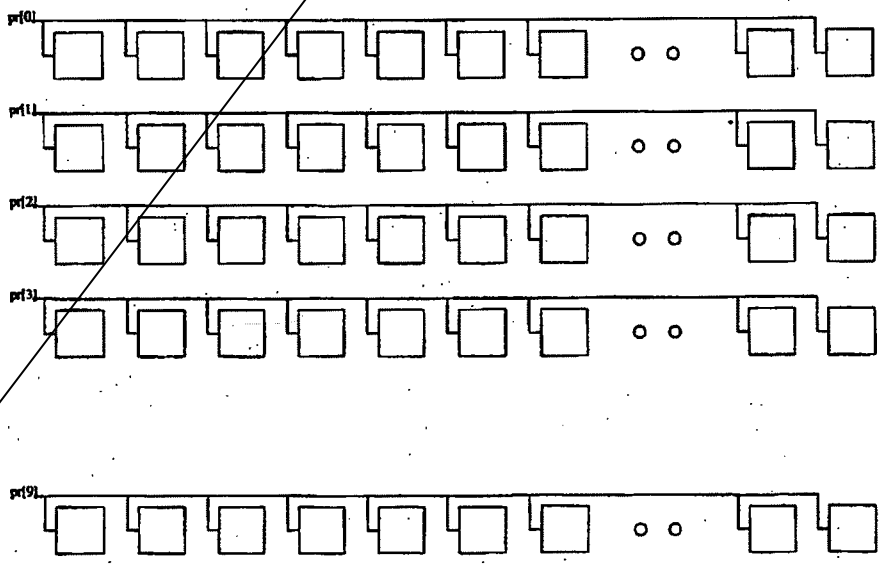


FIG. 382

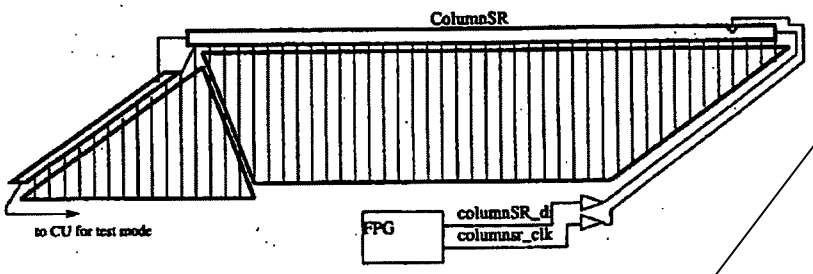


FIG. 383

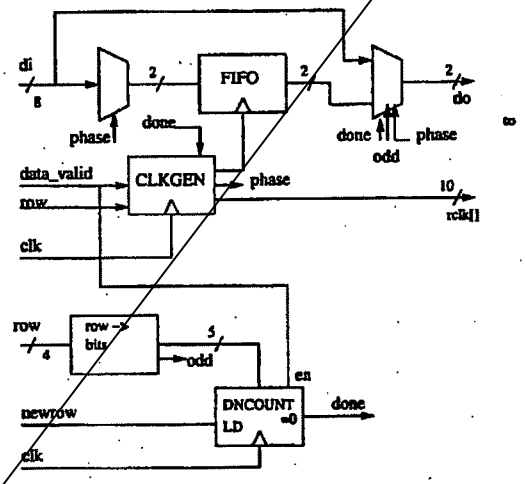


FIG. 384

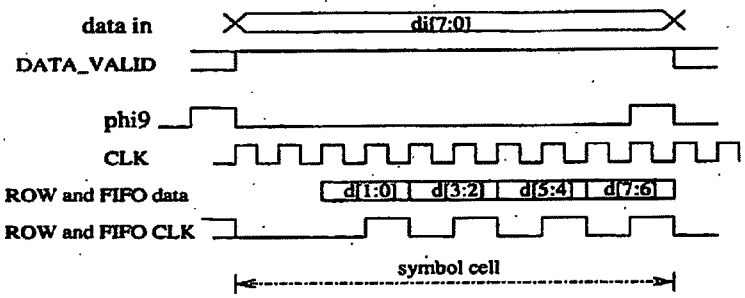


FIG. 385

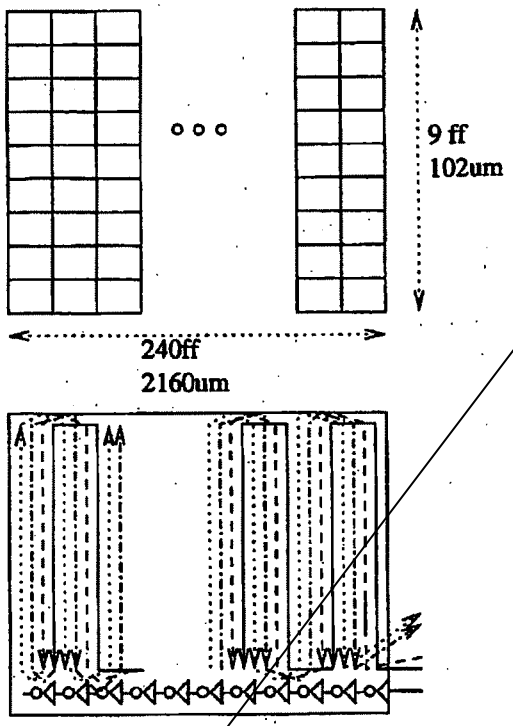


FIG. 386

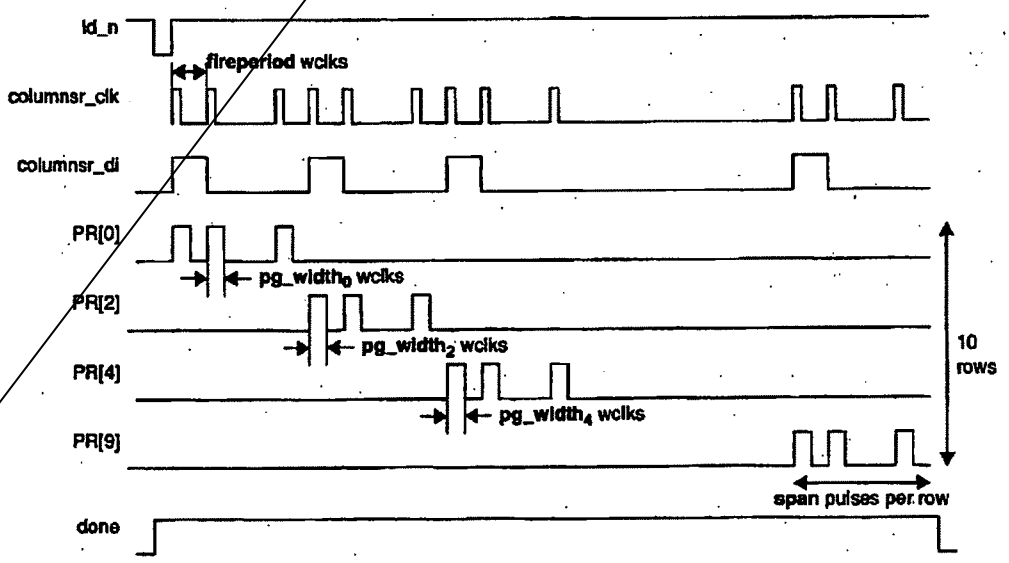


FIG. 387

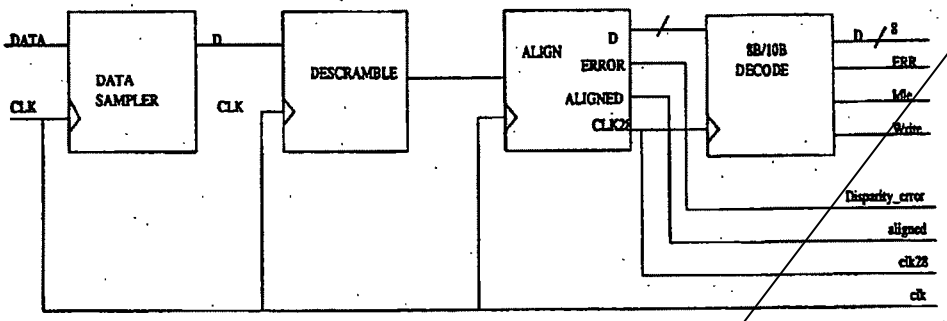


FIG. 388

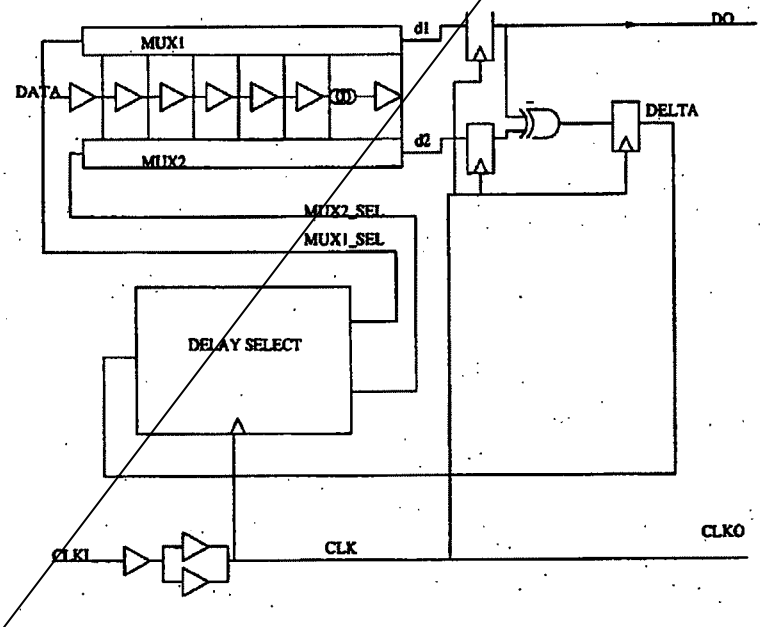


FIG. 389

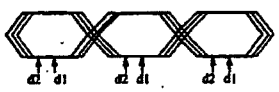


FIG. 390

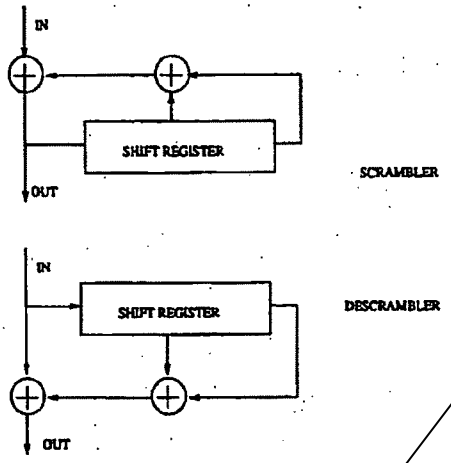


FIG. 391

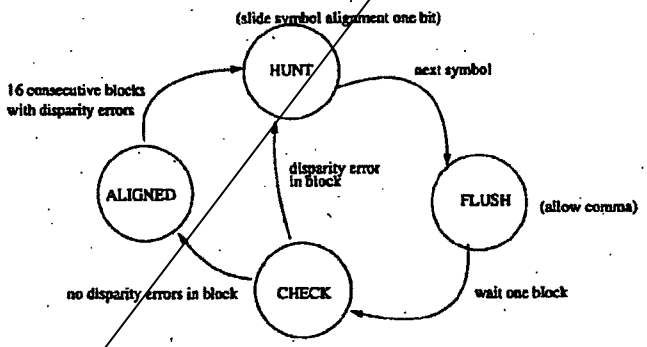


FIG. 392

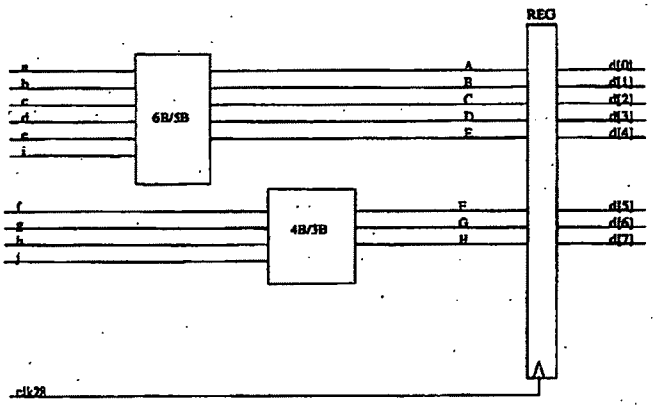


FIG. 393

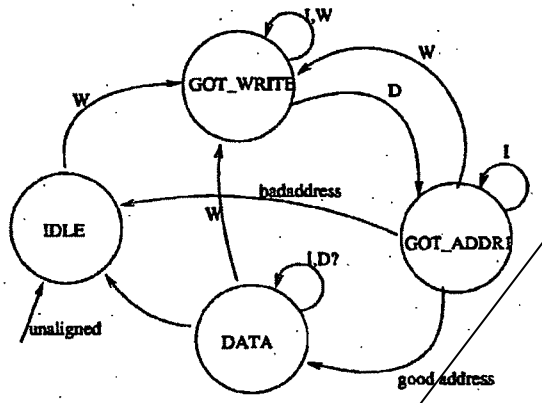


FIG. 394

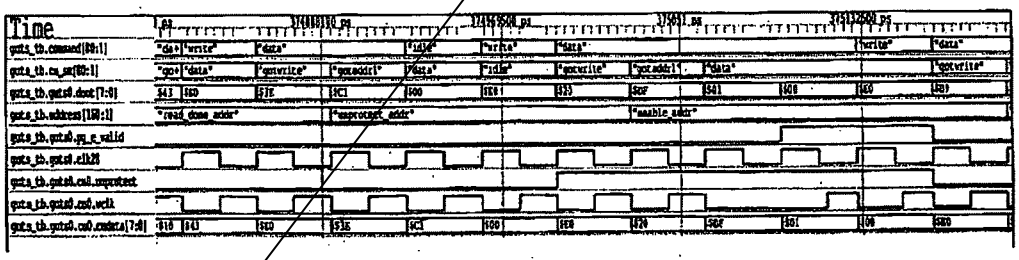


FIG. 395

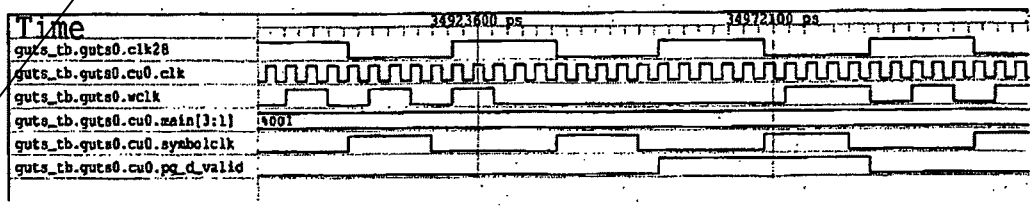


FIG. 396

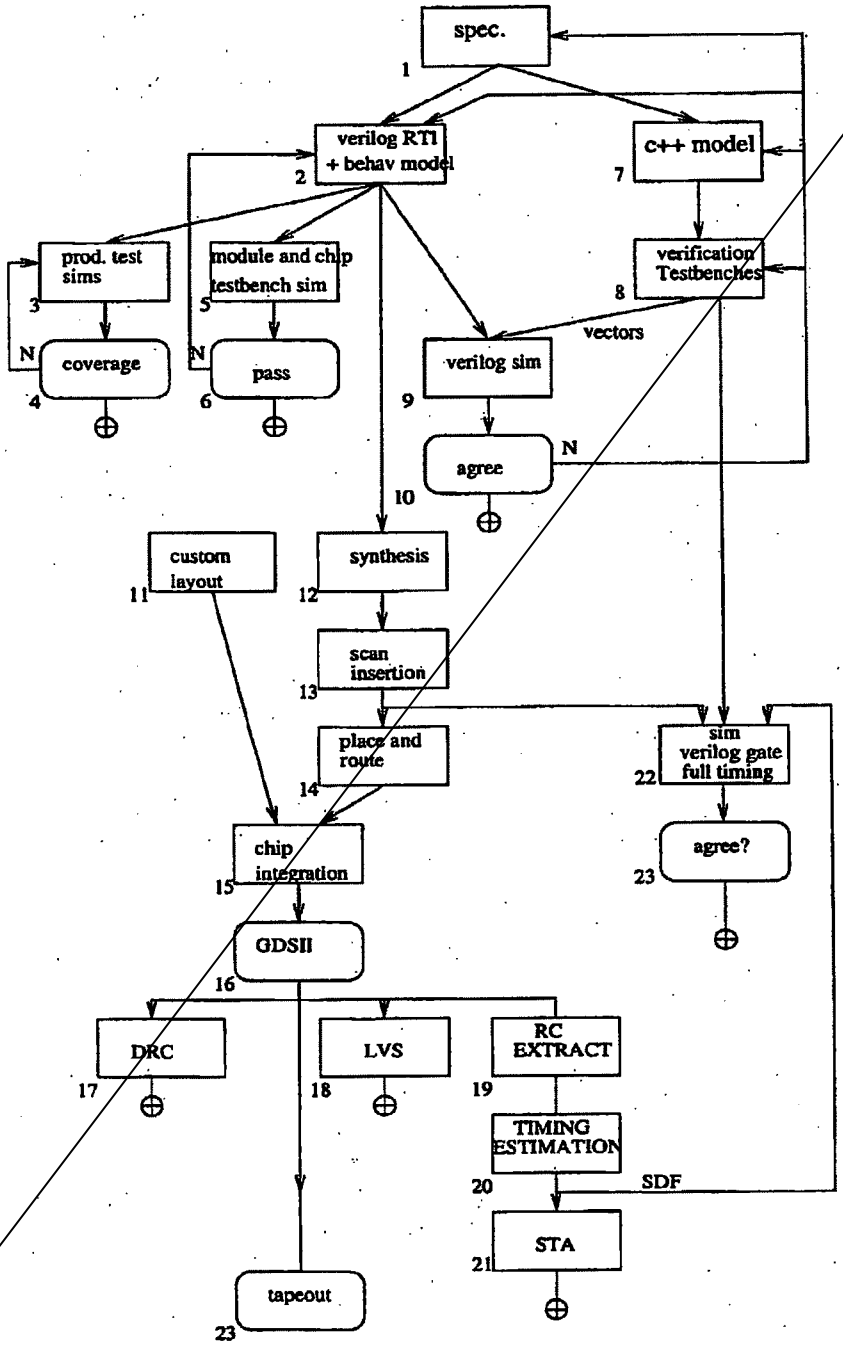

spec.
1
verilog RTl
+ behav model
2
c++ model
7
prod. test
sims
3
module and chip
testbench sim
5
verification
Testbenches
8
vectors
N
coverage
4
N
pass
6
verilog sim
9
agree
N
10
custom
layout
11
synthesis
12
scan
insertion
13
place and
route
14
sim
verilog gate
full timing
22
chip
integration
15
agree?
23
GDSII
16
DRC
17
LVS
18
RC
EXTRACT
19
TIMING
ESTIMATION
20
SDF
STA
21
tapeout
23

FIG. 397

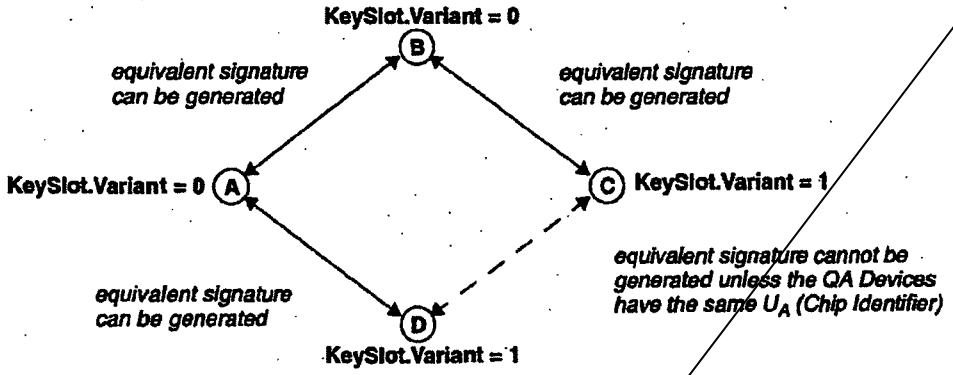


FIG. 398

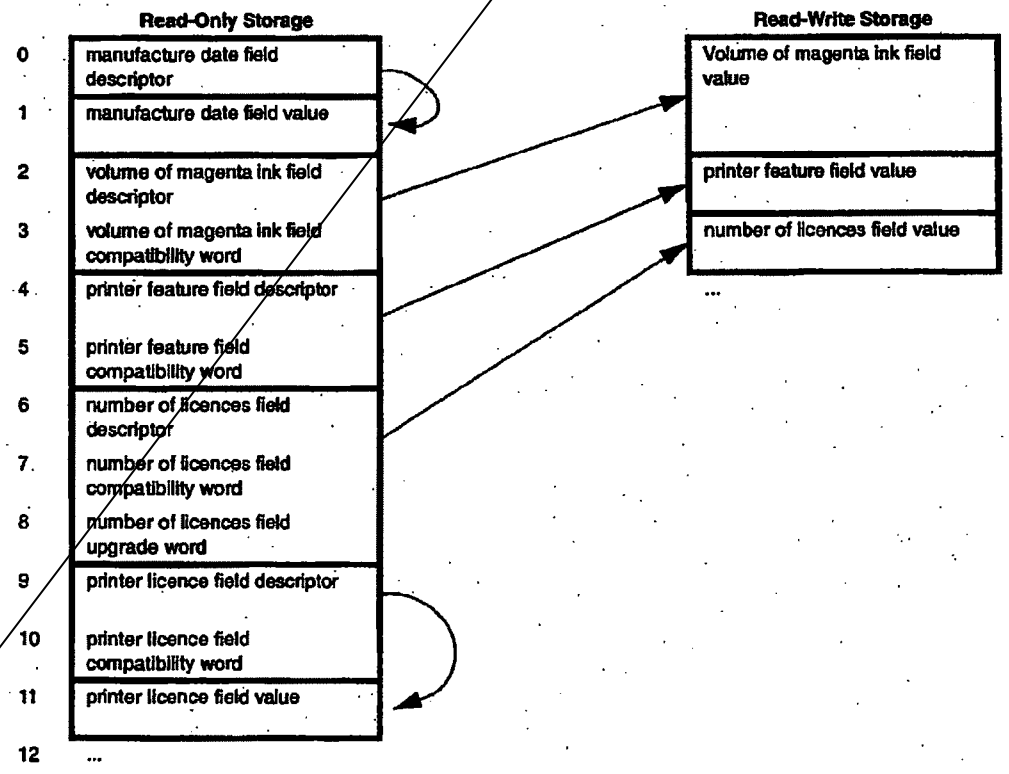


FIG. 399

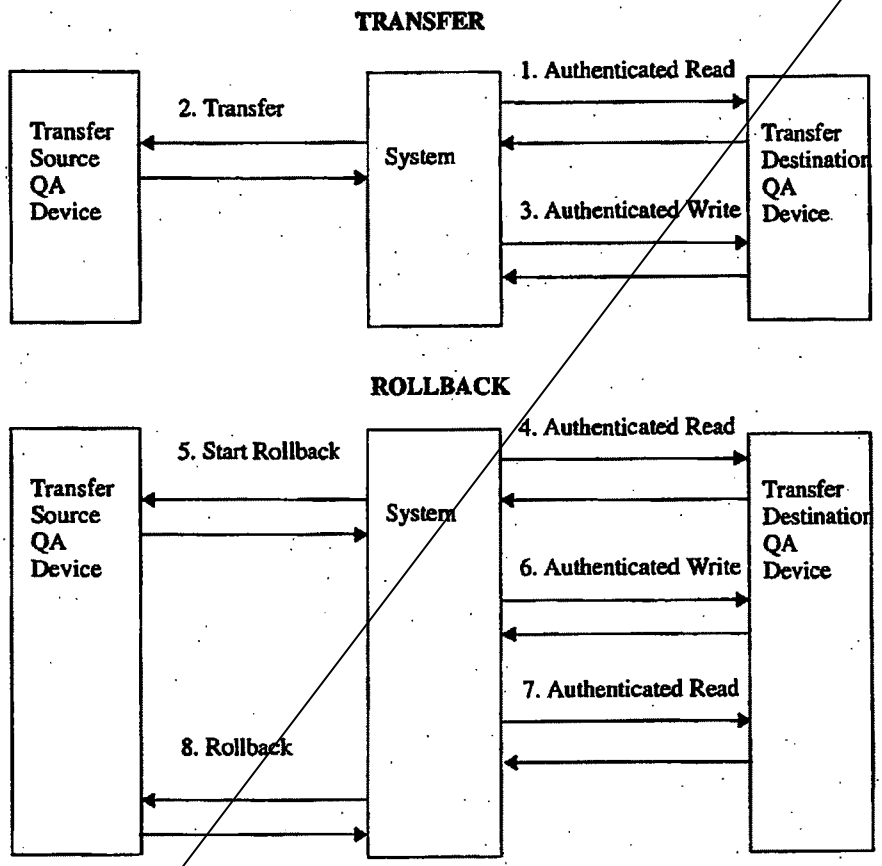

TRANSFER
Transfer Source QA Device
2. Transfer
System
1. Authenticated Read
3. Authenticated Write
Transfer Destination QA Device
ROLLBACK
Transfer Source QA Device
5. Start Rollback
System
4. Authenticated Read
6. Authenticated Write
7. Authenticated Read
8. Rollback
Transfer Destination QA Device

FIG. 400

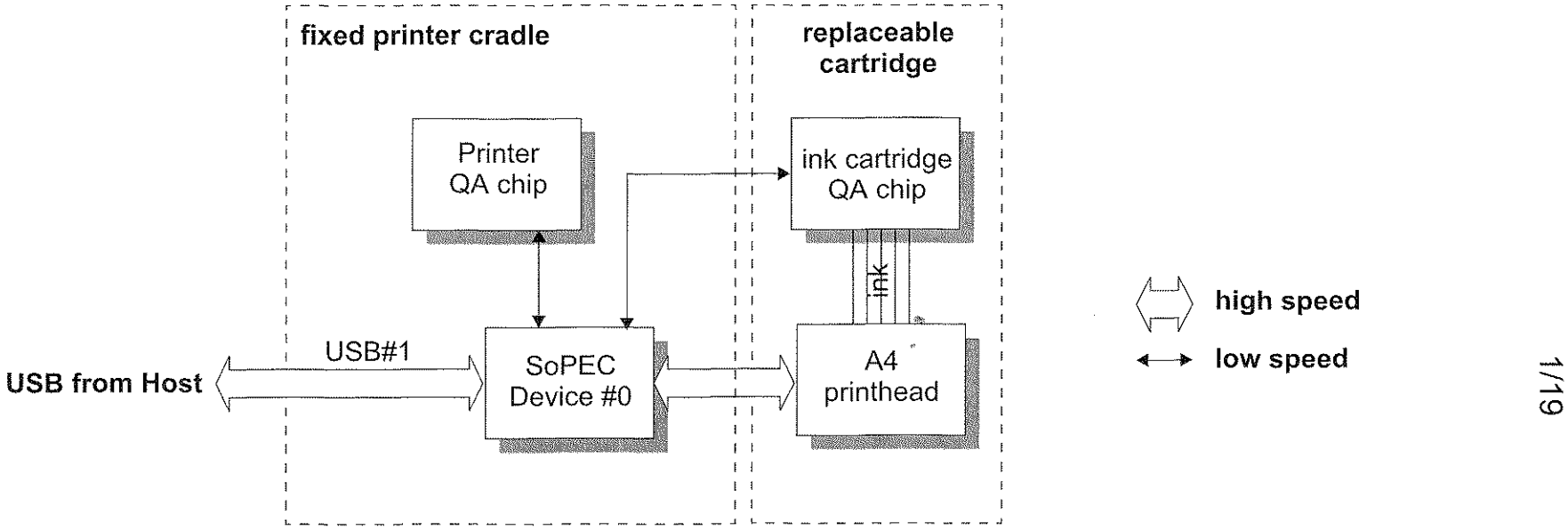

fixed printer cradle
replaceable cartridge
Printer QA chip
ink cartridge QA chip
ink
USB#1
USB from Host
SoPEC Device #0
A4 printhead
high speed
low speed

FIG. 1

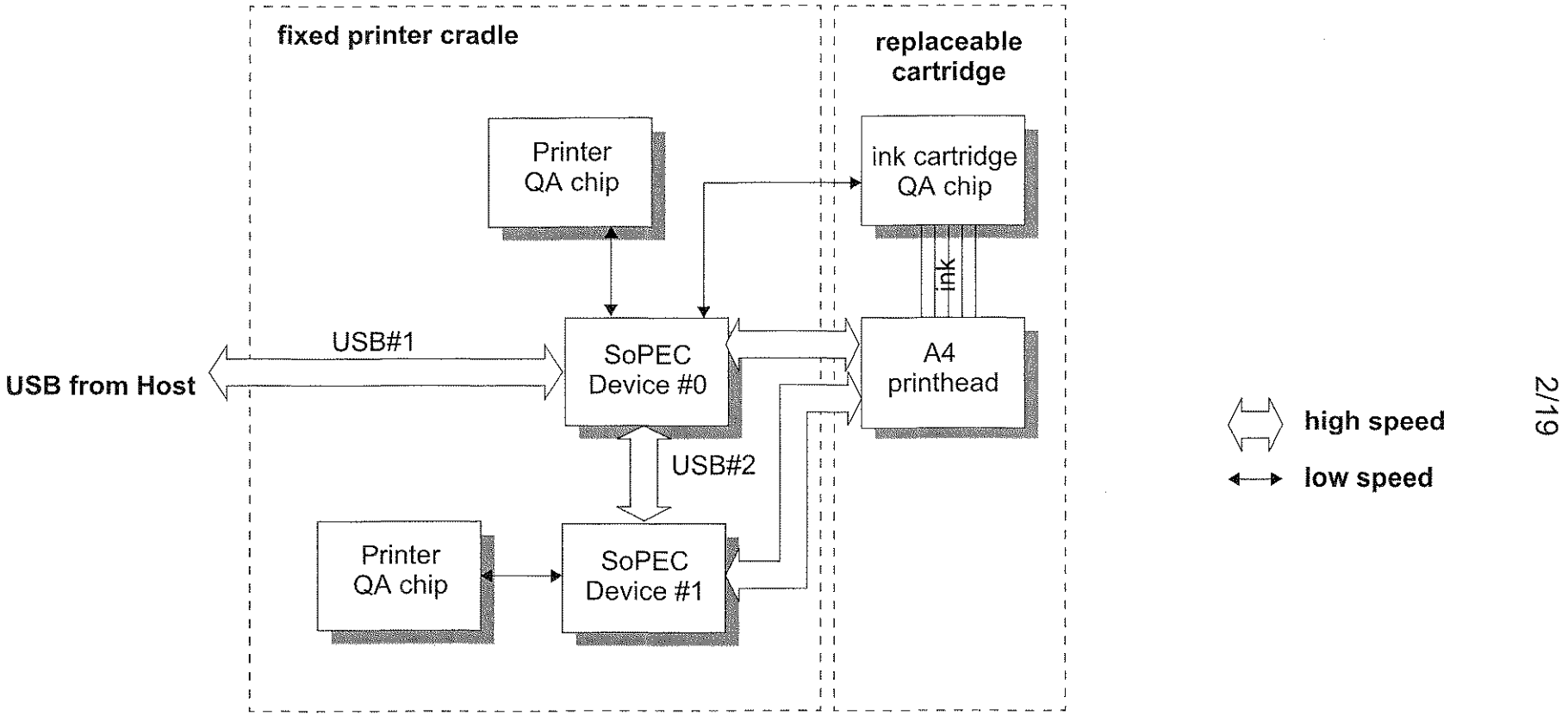

fixed printer cradle
replaceable cartridge
Printer QA chip
ink cartridge QA chip
ink
USB#1
USB from Host
SoPEC Device #0
A4 printhead
USB#2
Printer QA chip
SoPEC Device #1
high speed
low speed

FIG. 2

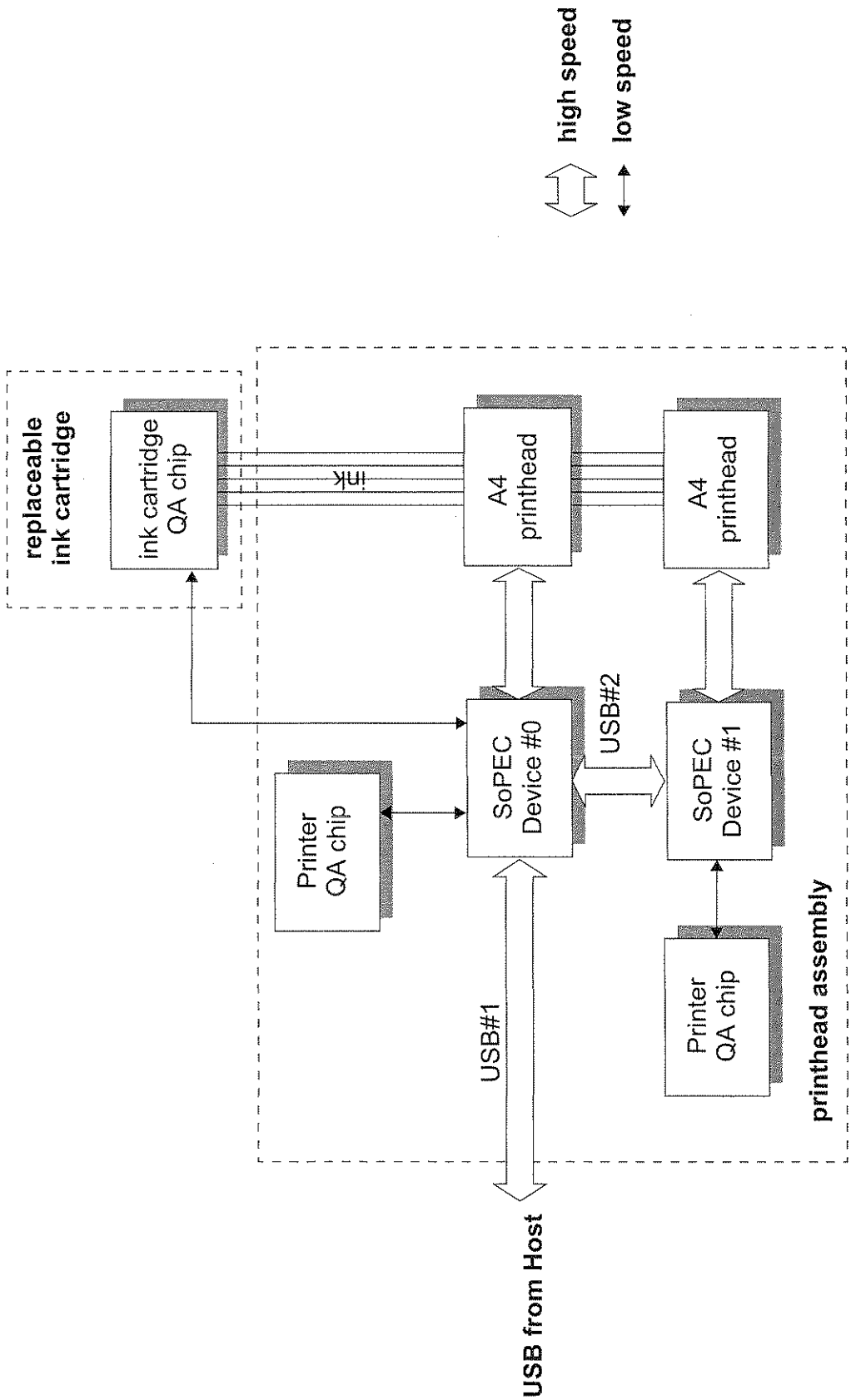

high speed
low speed
replaceable ink cartridge
ink cartridge QA chip
A4 printhead
A4 printhead
SoPEC Device #0
USB#2
SoPEC Device #1
Printer QA chip
Printer QA chip
USB#1
USB from Host
printhead assembly

FIG. 3

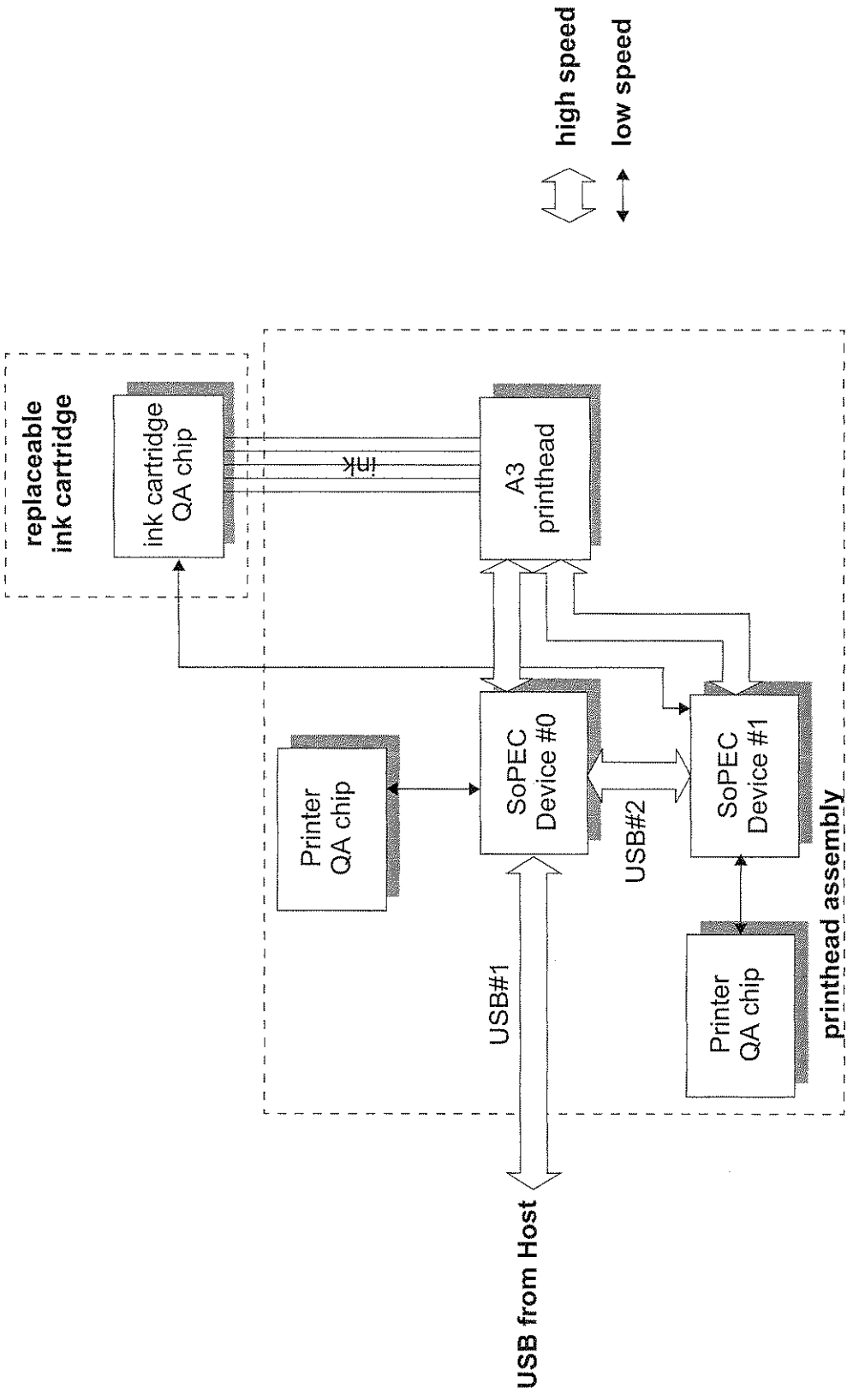

high speed
low speed
replaceable ink cartridge
ink cartridge QA chip
ink
A3 printhead
Printer QA chip
SoPEC Device #0
SoPEC Device #1
USB#2
USB#1
Printer QA chip
printhead assembly
USB from Host

FIG. 4

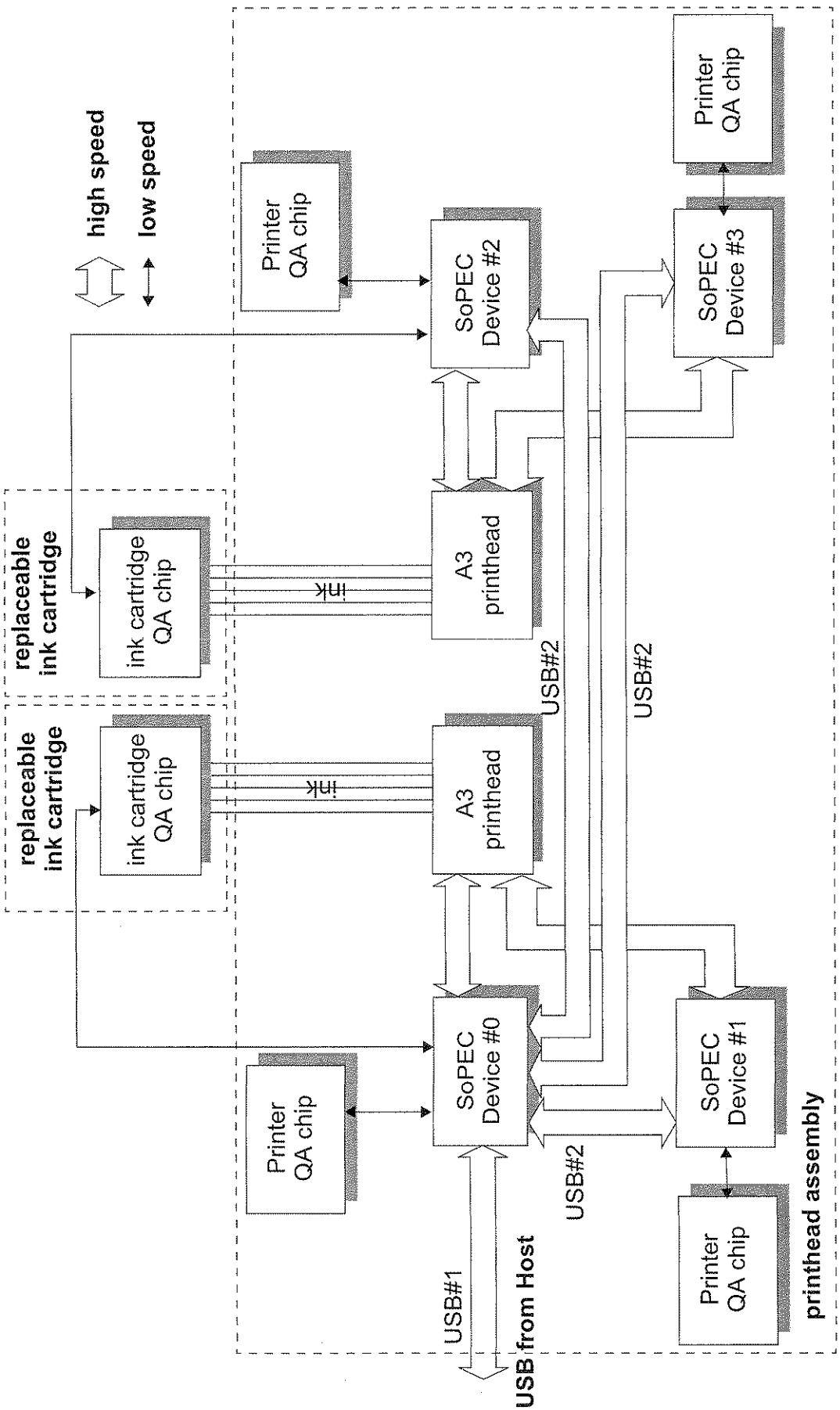

high speed
low speed
replaceable ink cartridge
ink cartridge QA chip
ink
A3 printhead
replaceable ink cartridge
ink cartridge QA chip
ink
A3 printhead
Printer QA chip
SoPEC Device #2
Printer QA chip
SoPEC Device #3
USB#2
USB#2
Printer QA chip
SoPEC Device #0
SoPEC Device #1
USB#2
Printer QA chip
USB#1
USB from Host
printhead assembly

FIG. 5

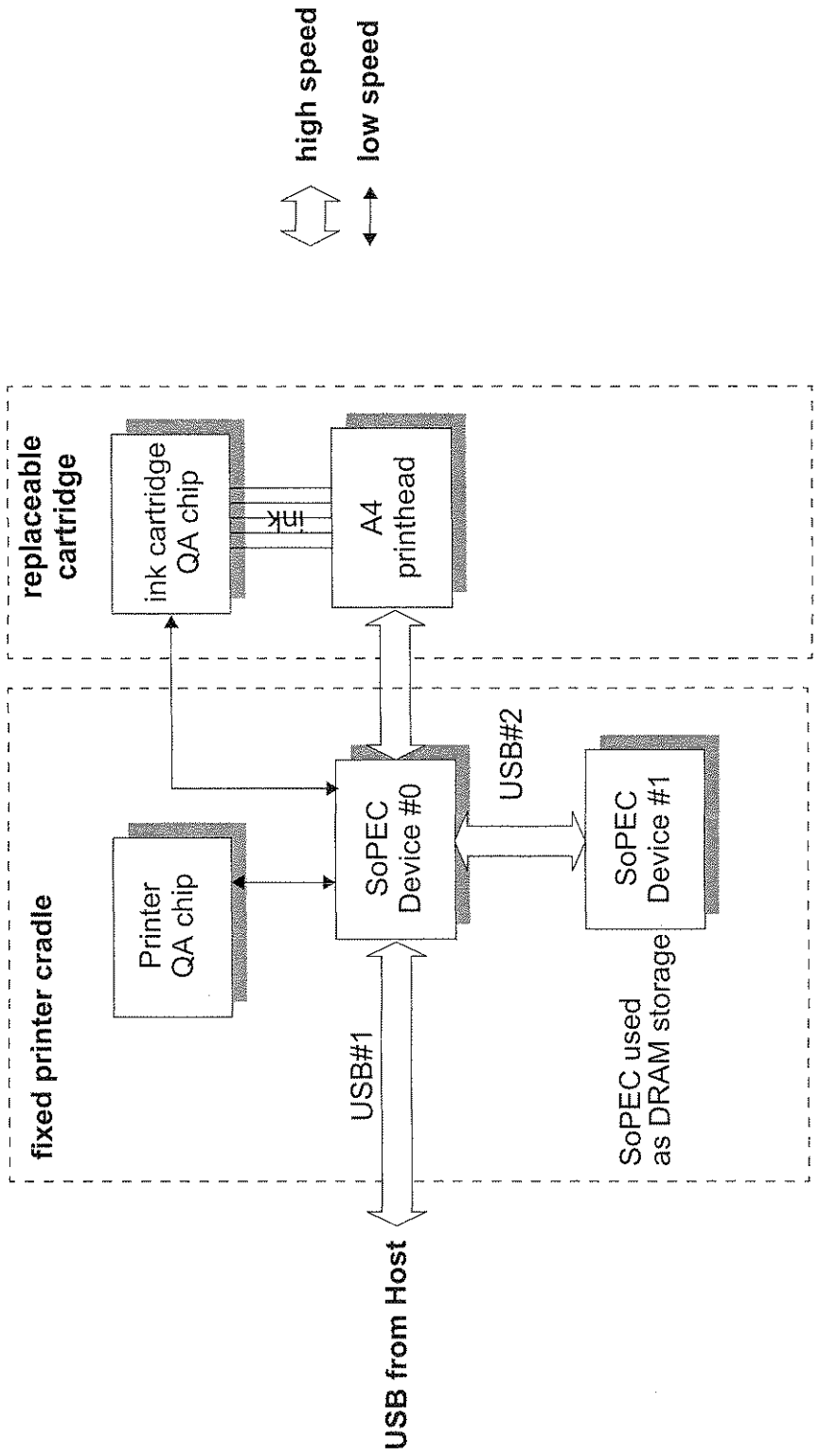

high speed
low speed
replaceable cartridge
ink cartridge QA chip
A4 printhead
fixed printer cradle
Printer QA chip
SoPEC Device #0
USB#2
SoPEC Device #1
USB#1
SoPEC used as DRAM storage
USB from Host

FIG. 6

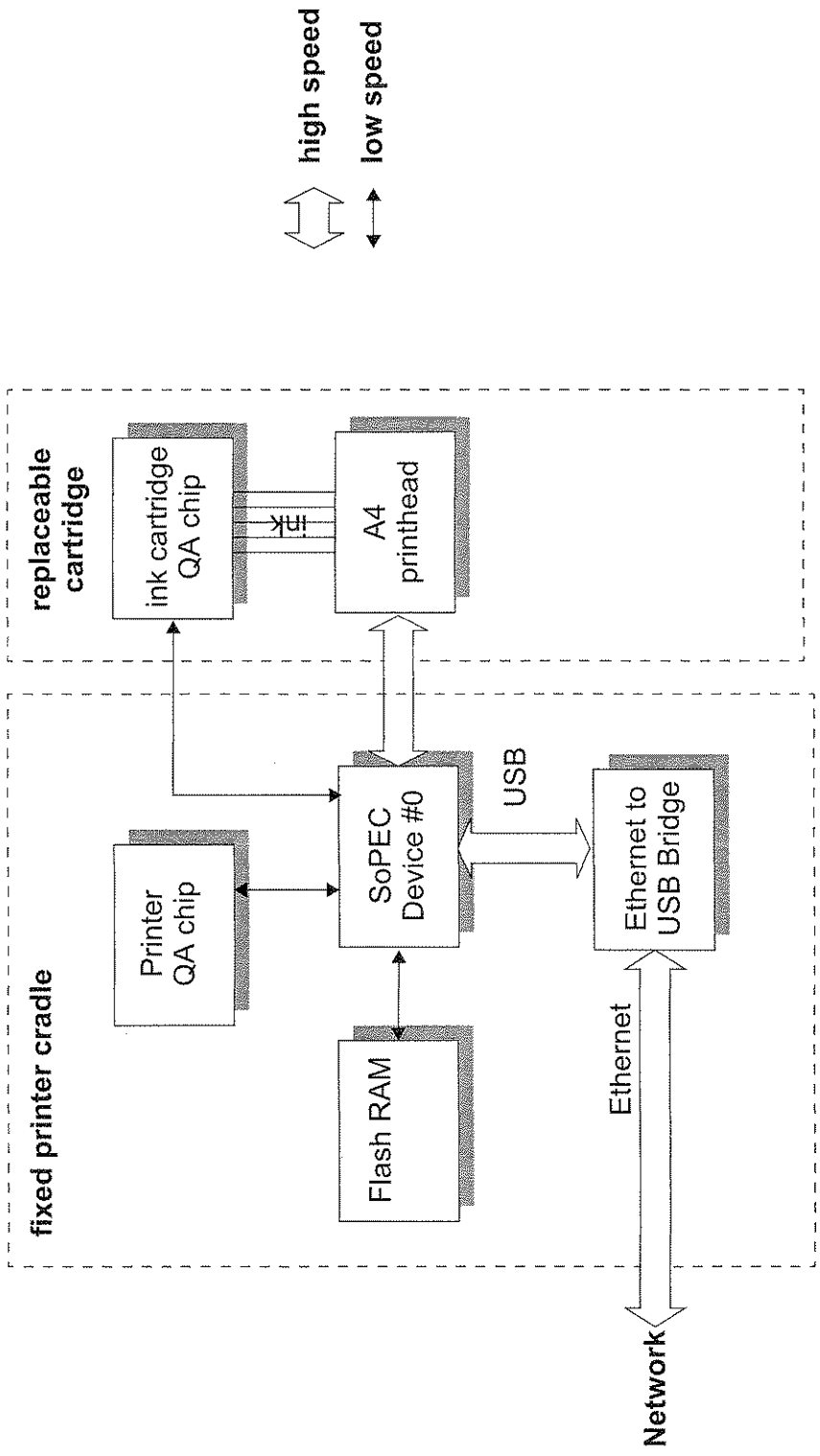

high speed
low speed
replaceable cartridge
ink cartridge QA chip
A4 printhead
fixed printer cradle
Printer QA chip
SoPEC Device #0
USB
Ethernet to USB Bridge
Flash RAM
Ethernet
Network

FIG. 7

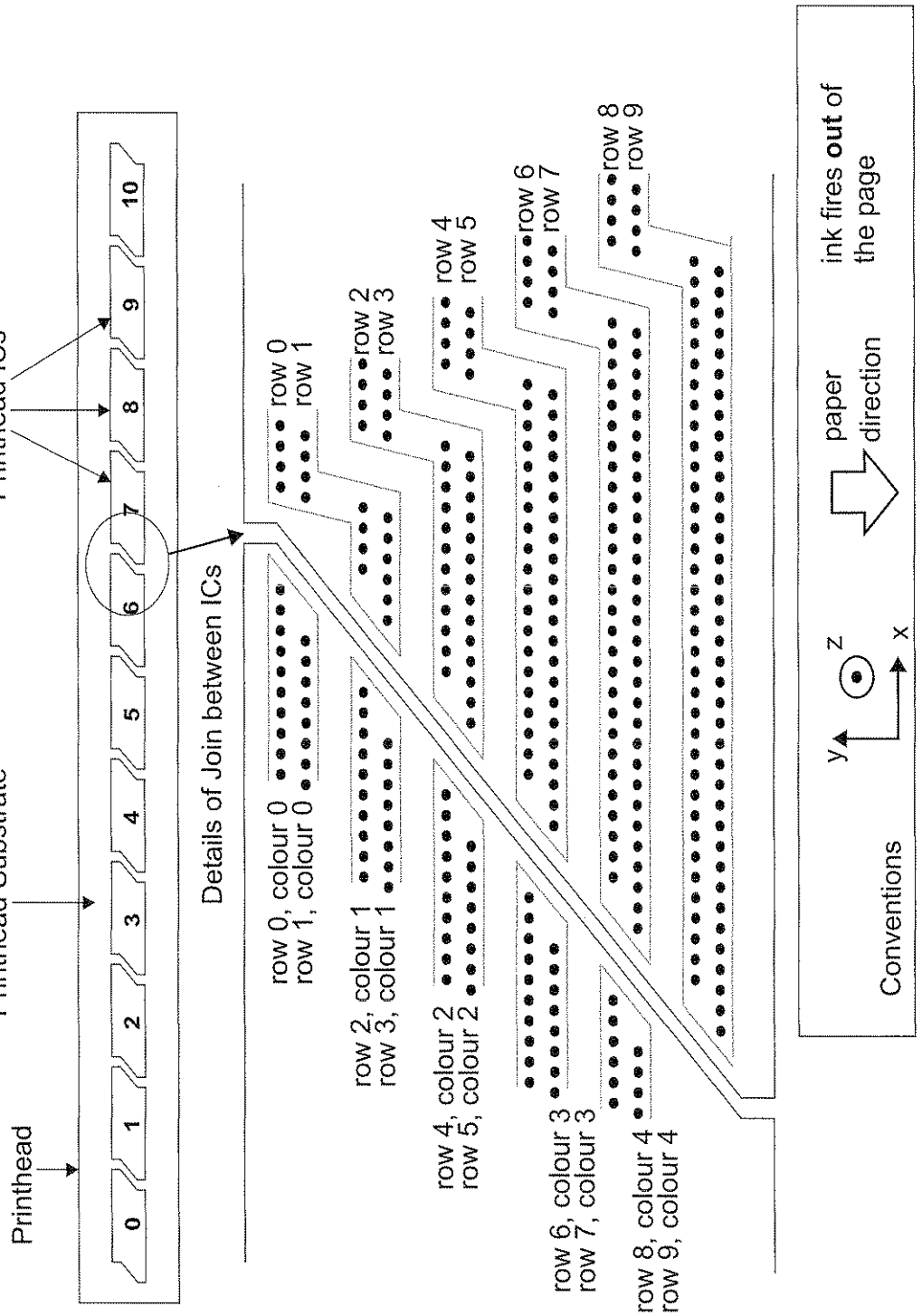

Printhead
Printhead Substrate
Printhead ICs
0
1
2
3
4
5
6
7
8
9
10
Details of Join between ICs
row 0, colour 0
row 1, colour 0
row 0
row 1
row 2, colour 1
row 3, colour 1
row 2
row 3
row 4, colour 2
row 5, colour 2
row 4
row 5
row 6, colour 3
row 7, colour 3
row 6
row 7
row 8, colour 4
row 9, colour 4
row 8
row 9
Conventions
y
z
x
paper direction
ink fires **out** of the page

FIG. 8

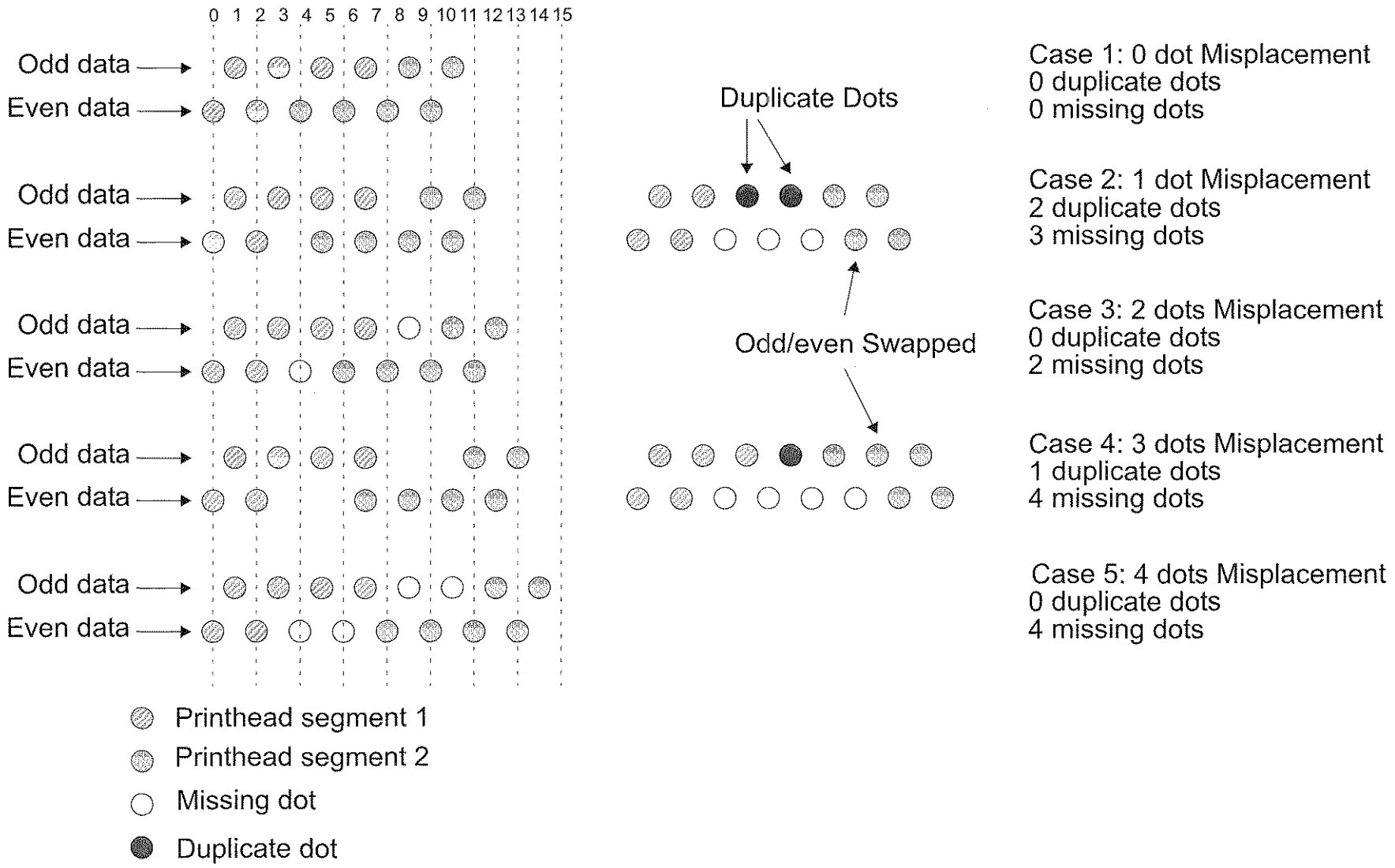

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Odd data
Even data
Odd data
Even data
Odd data
Even data
Odd data
Even data
Odd data
Even data
Printhead segment 1
Printhead segment 2
Missing dot
Duplicate dot
Duplicate Dots
Odd/even Swapped
Case 1: 0 dot Misplacement
0 duplicate dots
0 missing dots
Case 2: 1 dot Misplacement
2 duplicate dots
3 missing dots
Case 3: 2 dots Misplacement
0 duplicate dots
2 missing dots
Case 4: 3 dots Misplacement
1 duplicate dots
4 missing dots
Case 5: 4 dots Misplacement
0 duplicate dots
4 missing dots

FIG. 9

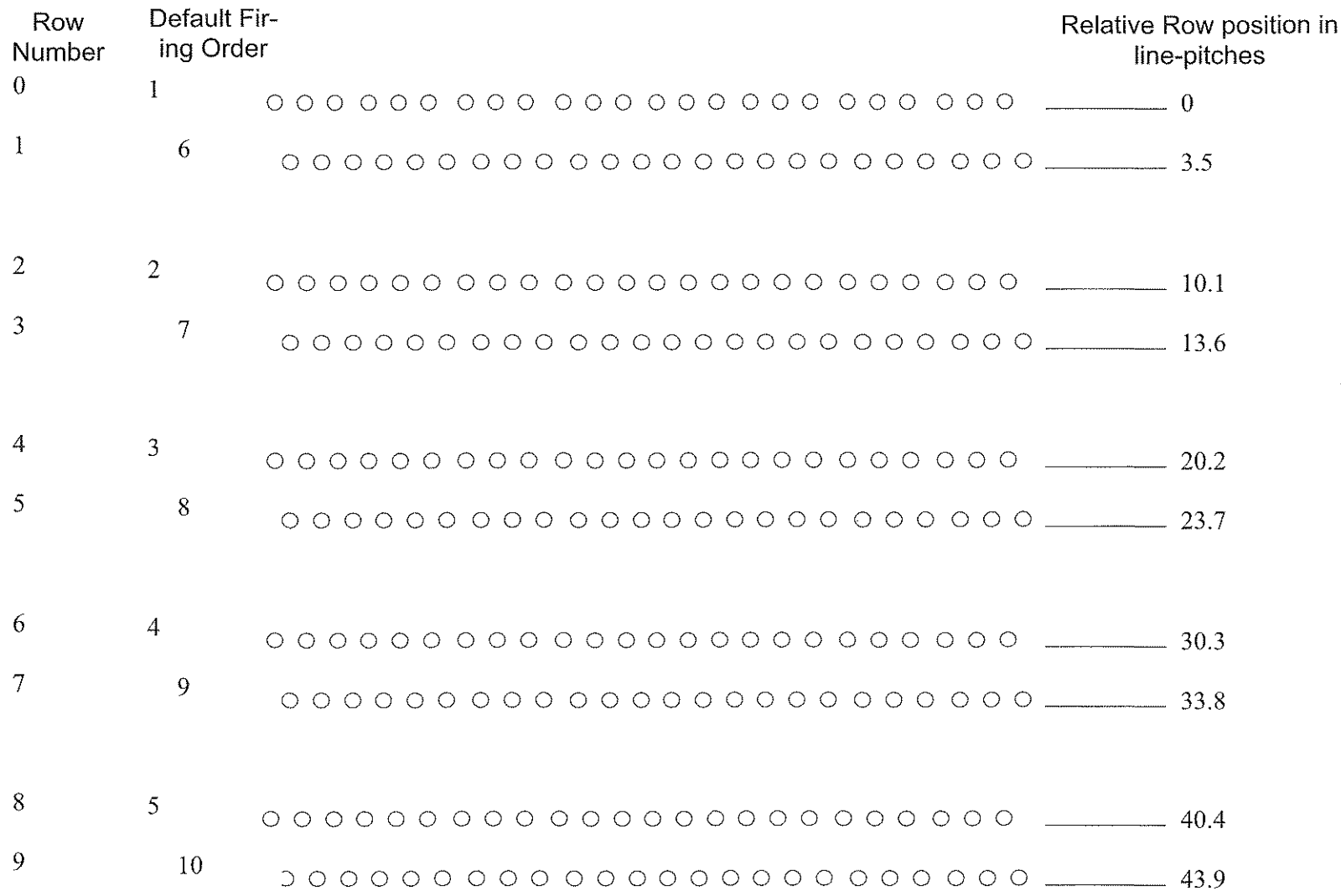

Row Number
Default Firing Order
Relative Row position in line-pitches
0
1
0
1
6
3.5
2
2
10.1
3
7
13.6
4
3
20.2
5
8
23.7
6
4
30.3
7
9
33.8
8
5
40.4
9
10
43.9

FIG. 10

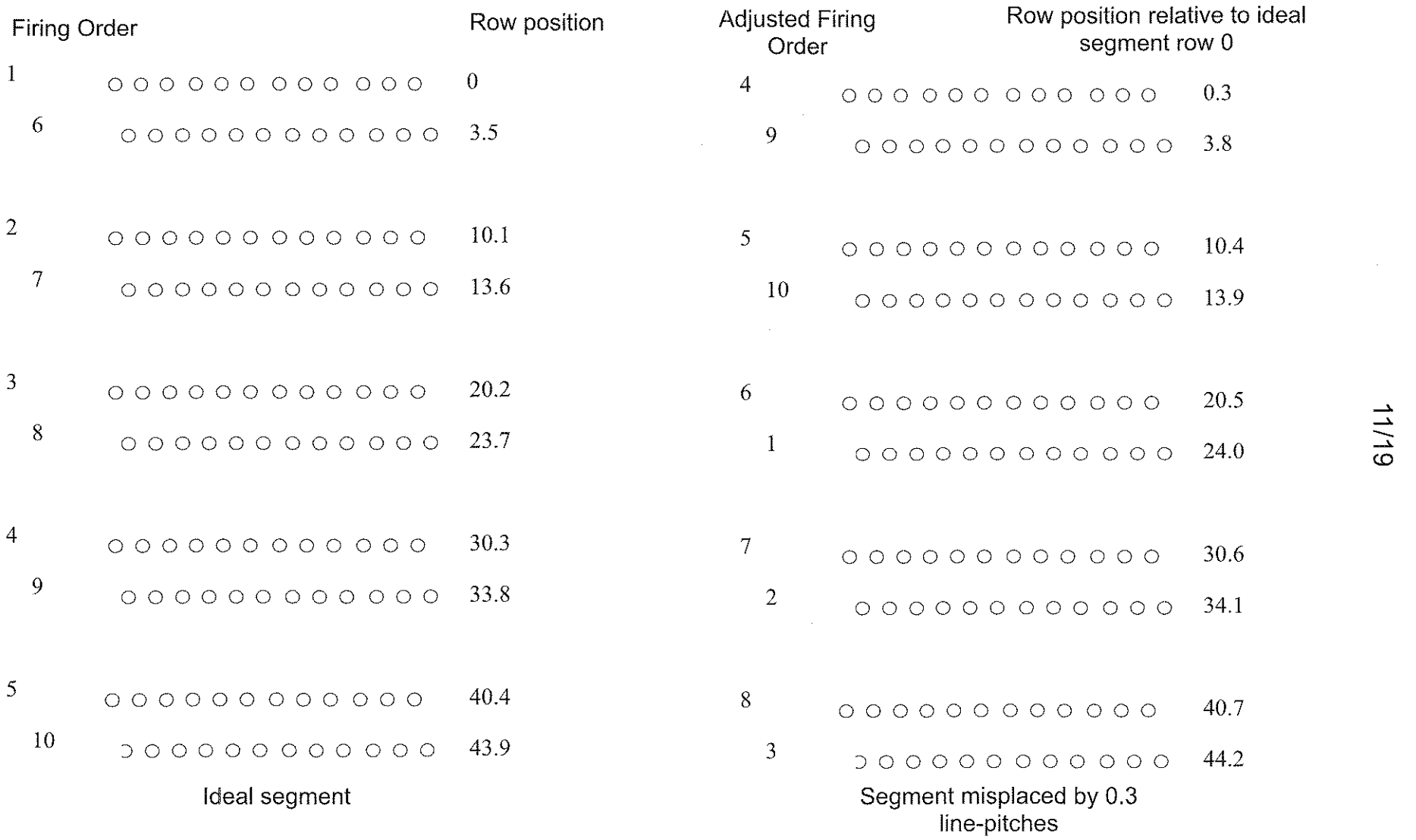

Firing Order
Row position
1
0
6
3.5
2
10.1
7
13.6
3
20.2
8
23.7
4
30.3
9
33.8
5
40.4
10
43.9
Ideal segment
Adjusted Firing Order
Row position relative to ideal segment row 0
4
0.3
9
3.8
5
10.4
10
13.9
6
20.5
1
24.0
7
30.6
2
34.1
8
40.7
3
44.2
Segment misplaced by 0.3 line-pitches

FIG. 11

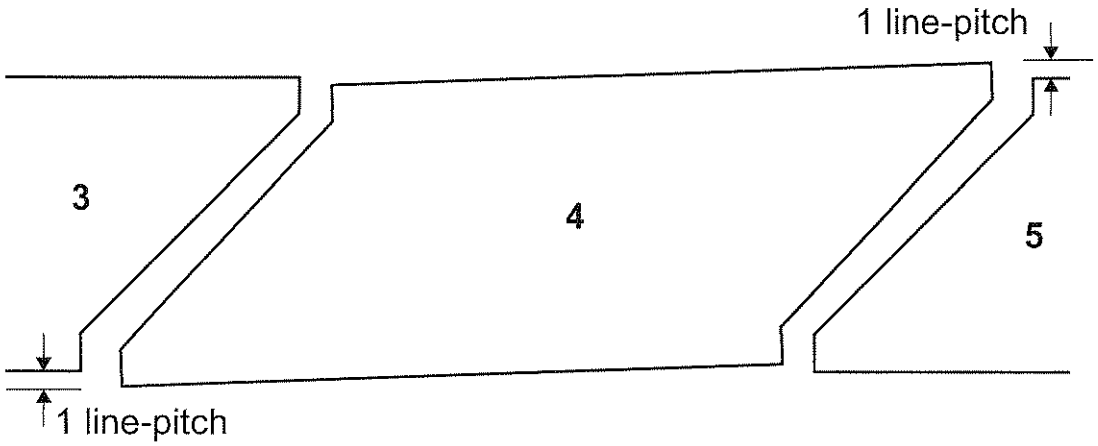

1 line-pitch
3
4
5
1 line-pitch

FIG. 12

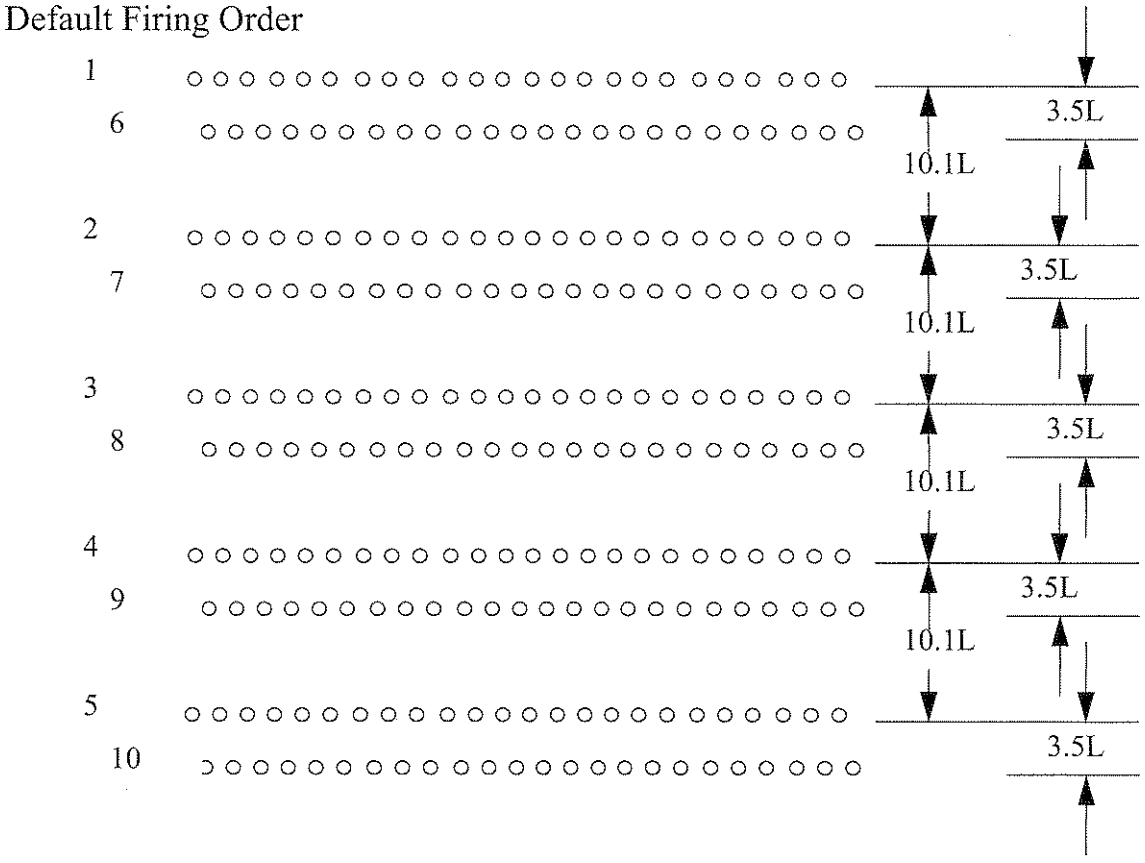

Default Firing Order
1
6
2
7
3
8
4
9
5
10
10.1L
3.5L

FIG. 13

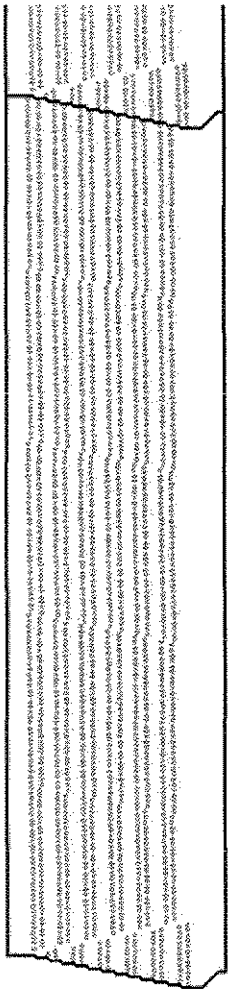


FIG. 14



FIG. 15

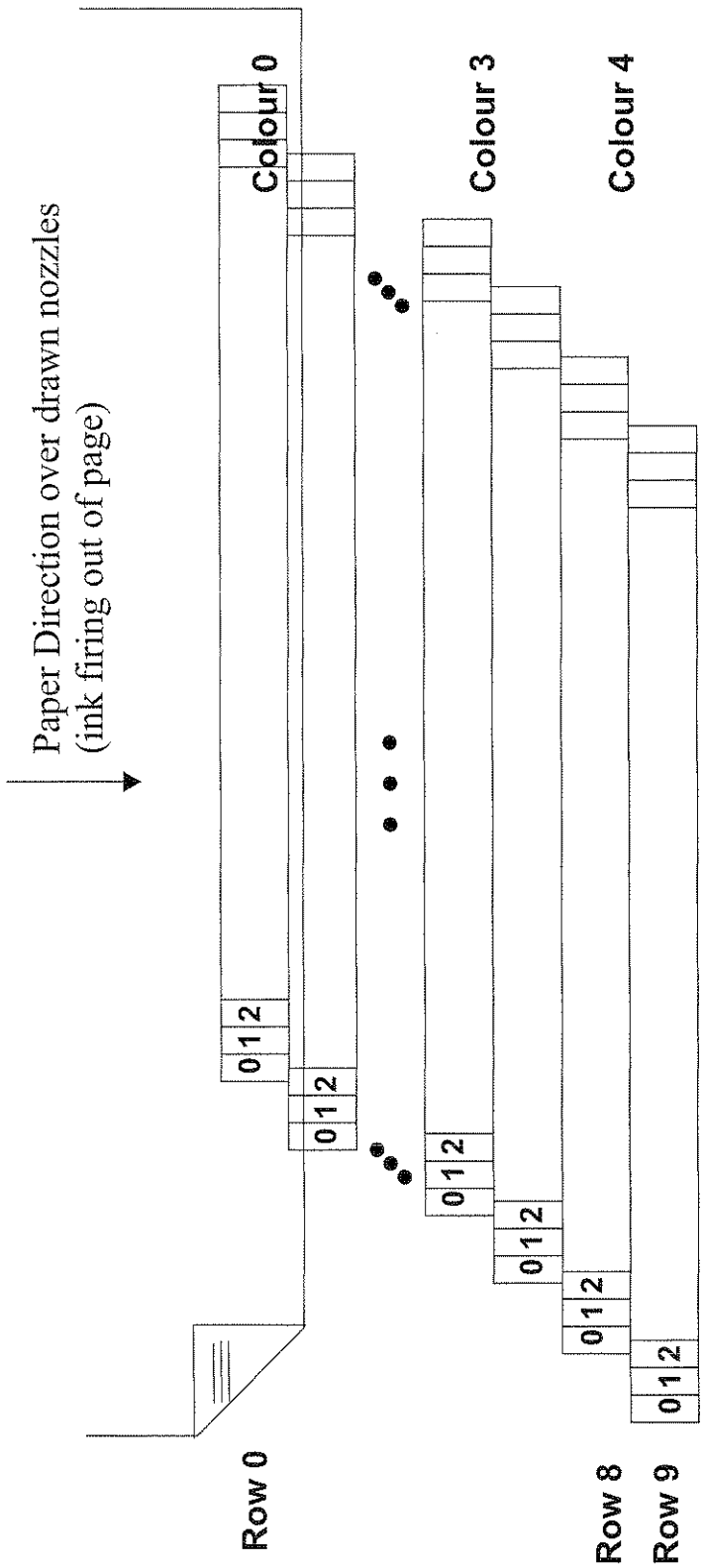

Paper Direction over drawn nozzles
(ink firing out of page)
Colour 0
Colour 3
Colour 4
0 1 2
Row 0
Row 8
Row 9

FIG. 16

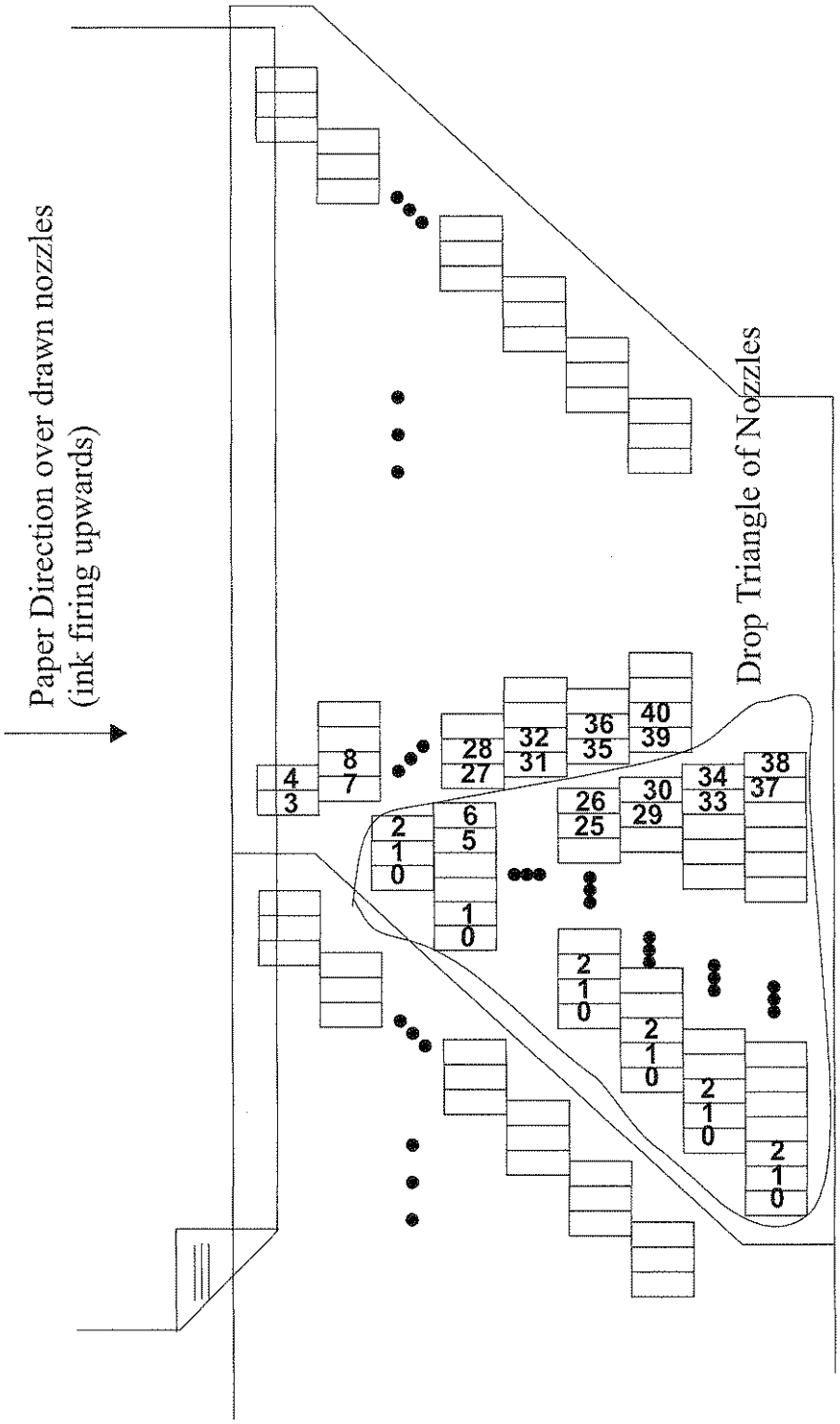

Paper Direction over drawn nozzles
(ink firing upwards)
Drop Triangle of Nozzles
8
4
7
3
28
27
32
31
36
35
40
39
38
37
34
33
30
29
26
25
6
5
2
1
0
1
0
2
1
0
2
1
0
2
1
0
2
1
0

FIG. 17

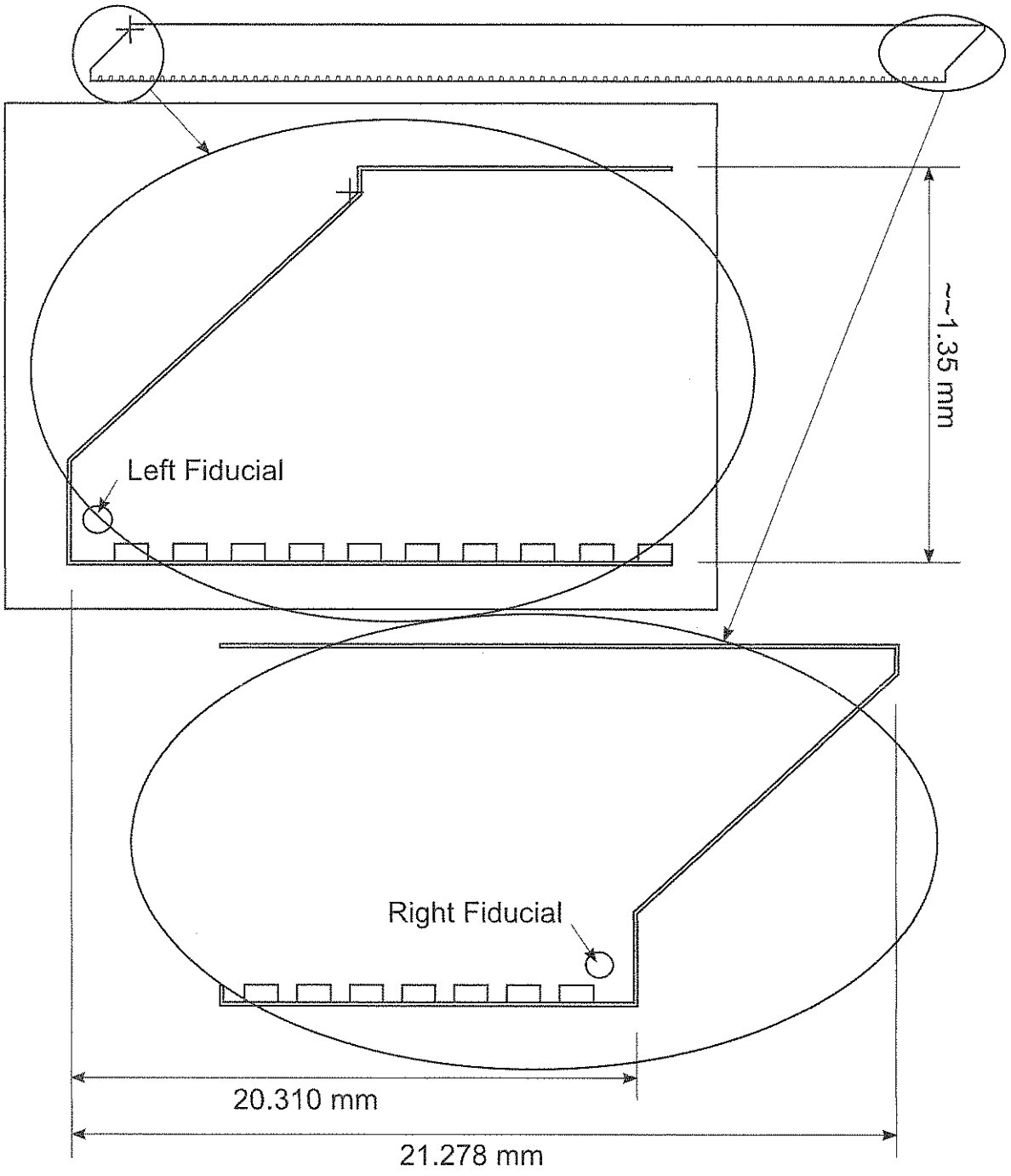


FIG. 18

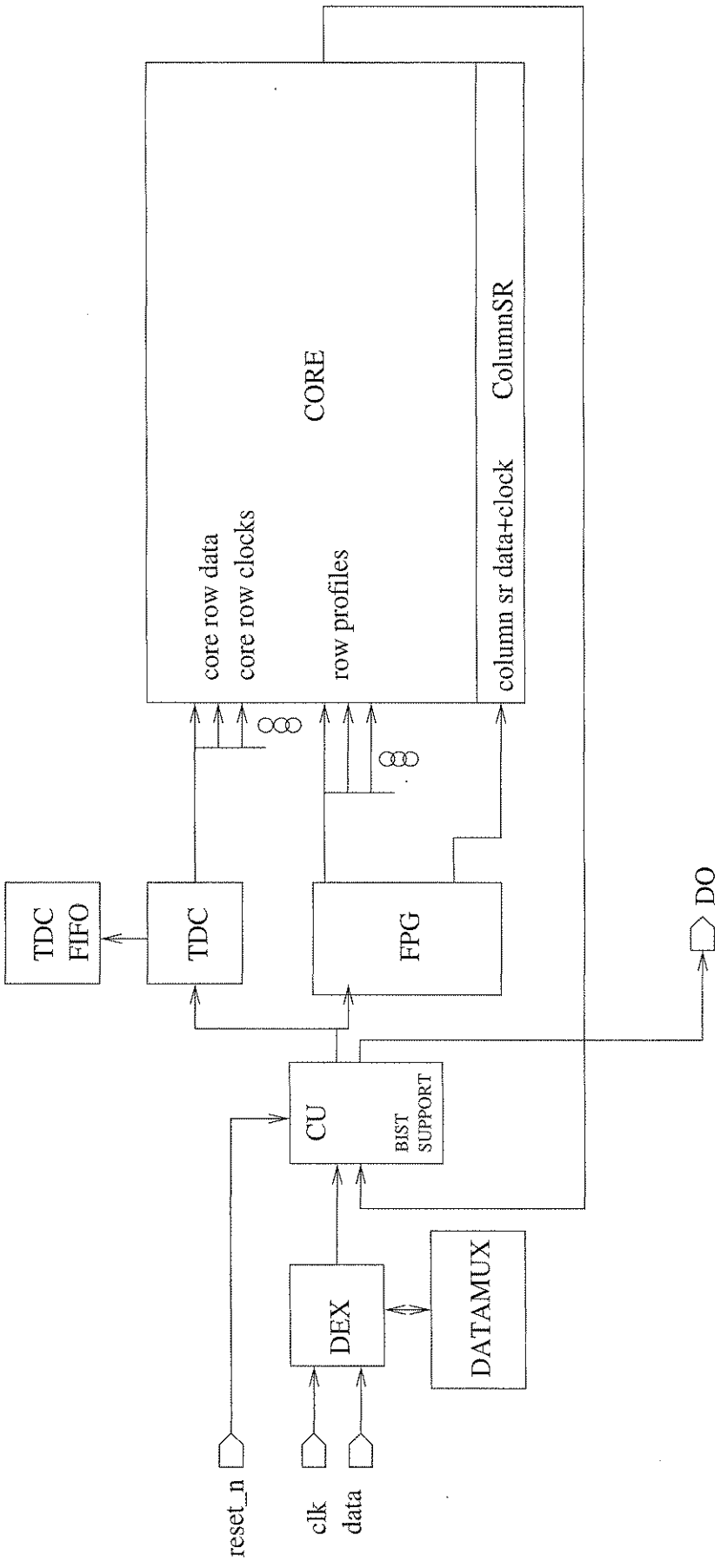

reset_n
clk
data
DEX
DATAMUX
CU
BIST
SUPPORT
TDC
FIFO
TDC
FPG
DO
CORE
core row data
core row clocks
row profiles
column sr data+clock
ColumnSR
8
8

FIG. 19

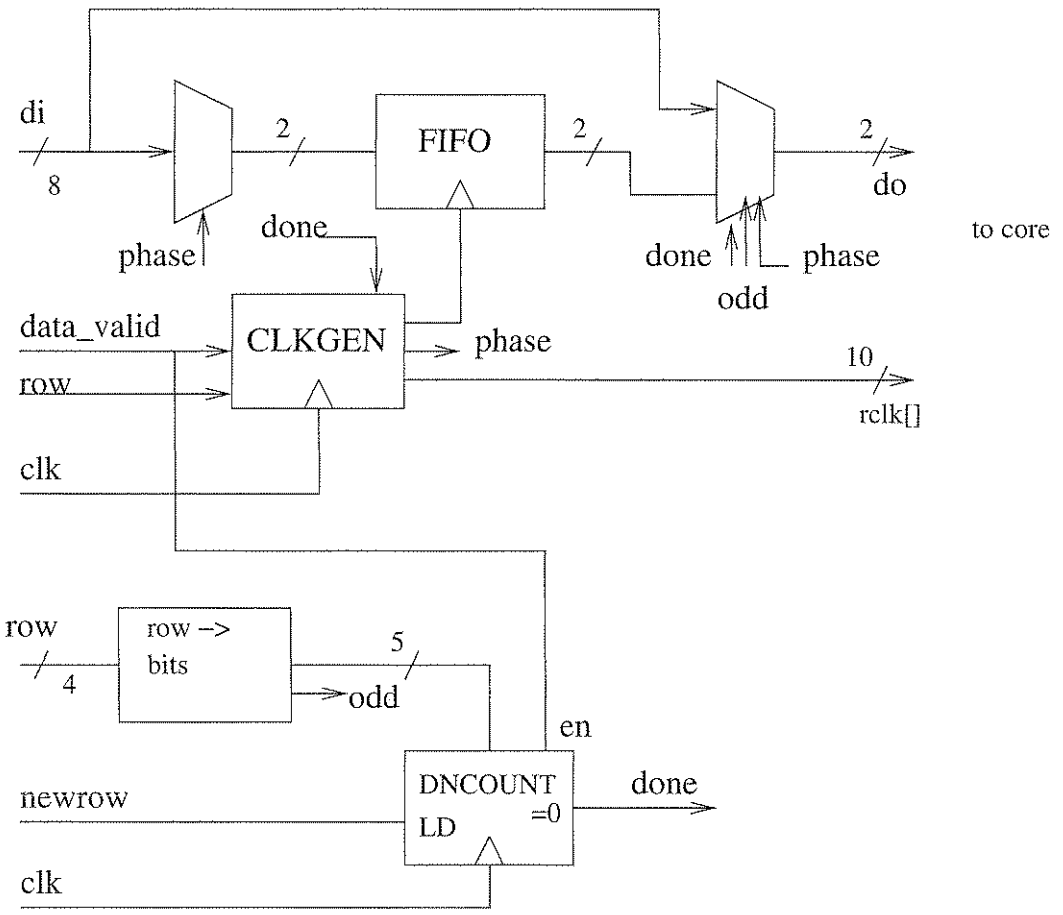

di
8
2
FIFO
2
2
do
to core
phase
done
done
odd
phase
data_valid
row
clk
CLKGEN
phase
10
rclk[]
row
4
row ->
bits
5
odd
en
newrow
clk
DNCOUNT
LD
=0
done

FIG. 20

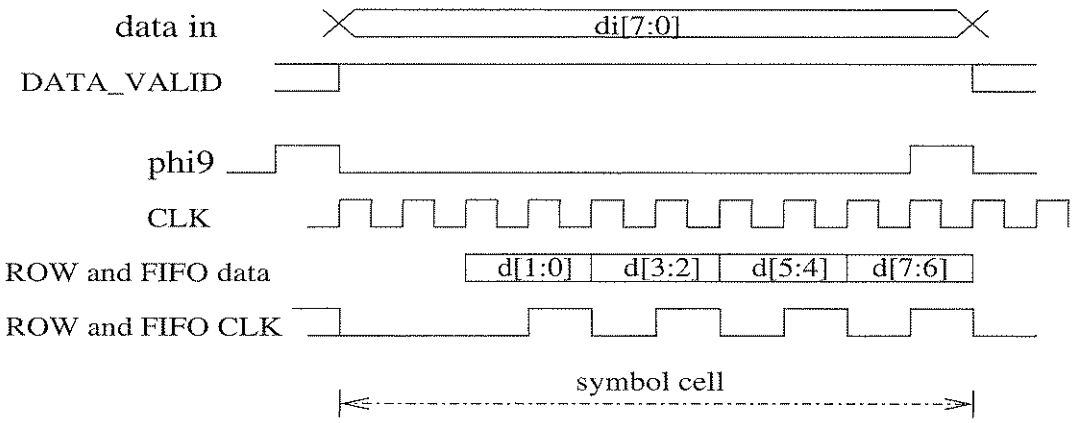

data in
di[7:0]
DATA_VALID
phi9
CLK
ROW and FIFO data
d[1:0]
d[3:2]
d[5:4]
d[7:6]
ROW and FIFO CLK
symbol cell

FIG. 21

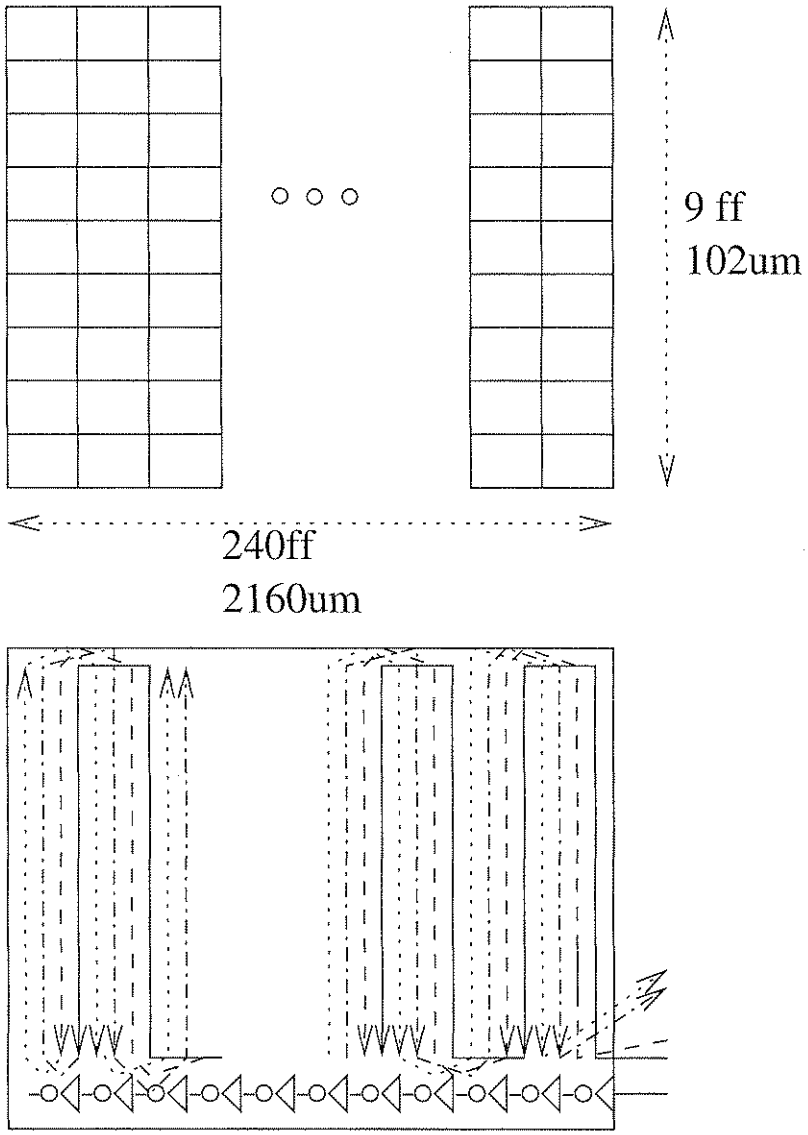


FIG. 22