

SoC Robot – Verilog source Code

System Design Innovation & Application Research Center

- I. HDL
- II. FPGA
- III. Image Processing
- IV. Verilog Code 분석



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



HDL = Hardware Description language

VHDL

- 미 국방성(Department of Defense)의 요구에 따라 연구되기 시작
- 1993표준화 (IEEE 1164)
- 무기체제에 채용되는 전자장비의 개선과 유지보수의 큰 부담을 줄이고자 개발

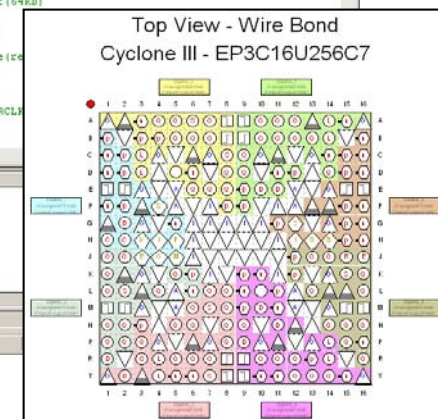
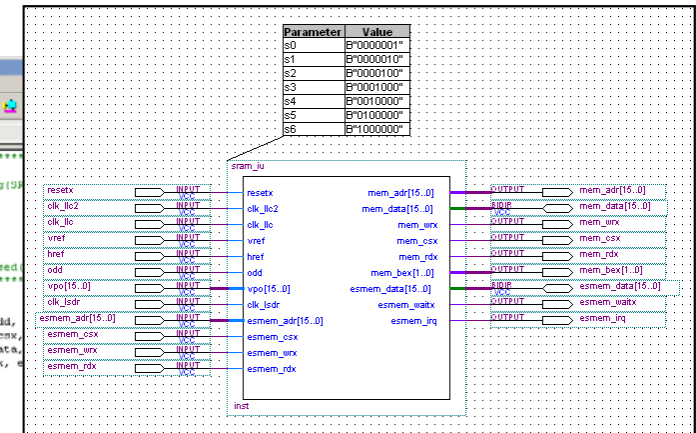
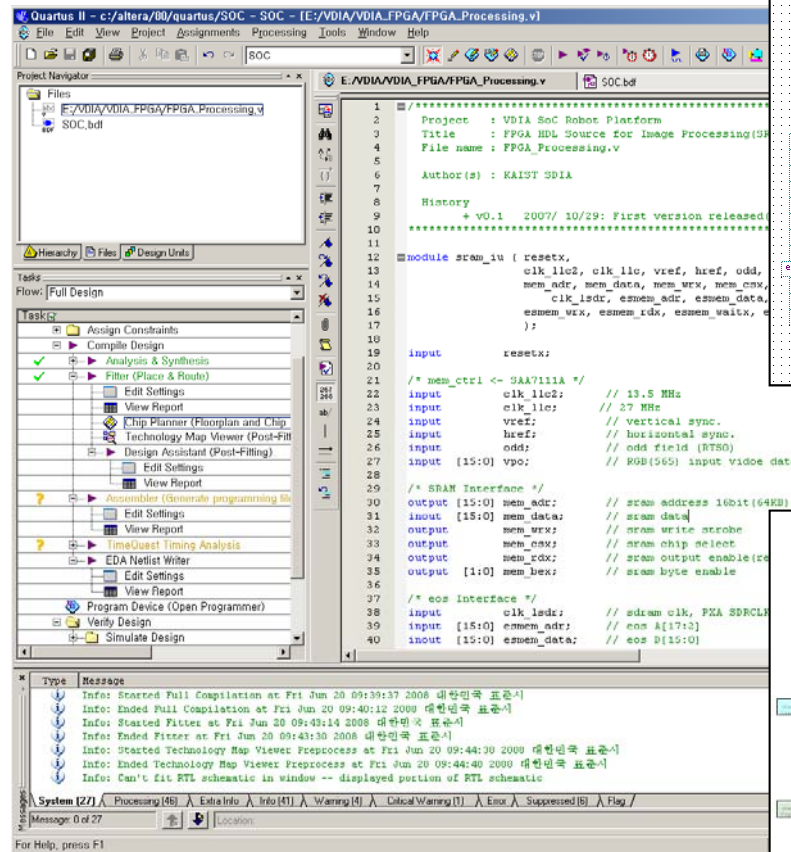
Verilog-HDL

- Gateway Design Automation 회사에서 개발
- 1985년에 시뮬레이터 제품으로 시장에 발표
- C언어와 유사



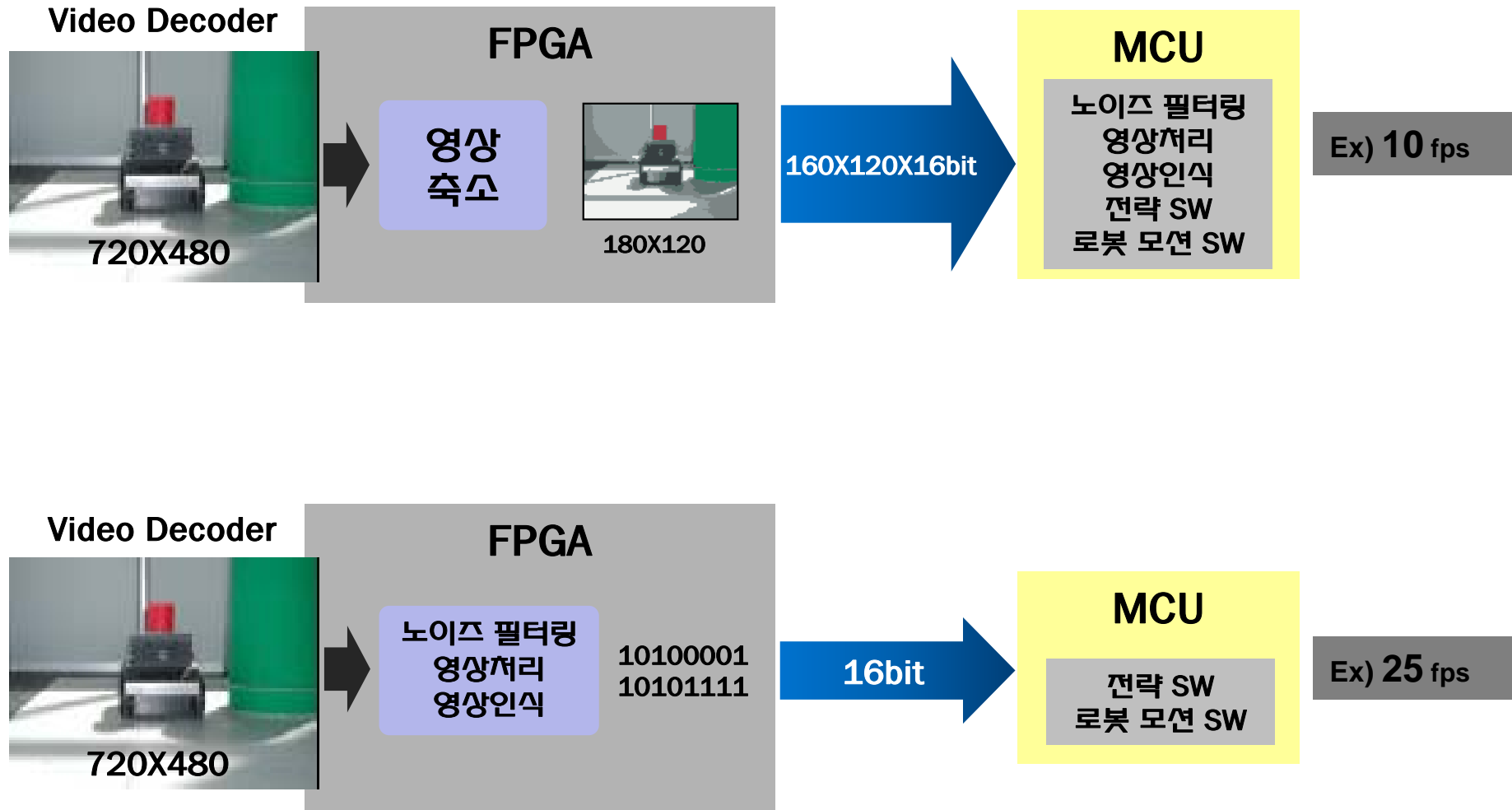
FPGA = Field Programmable Gate Array

Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!

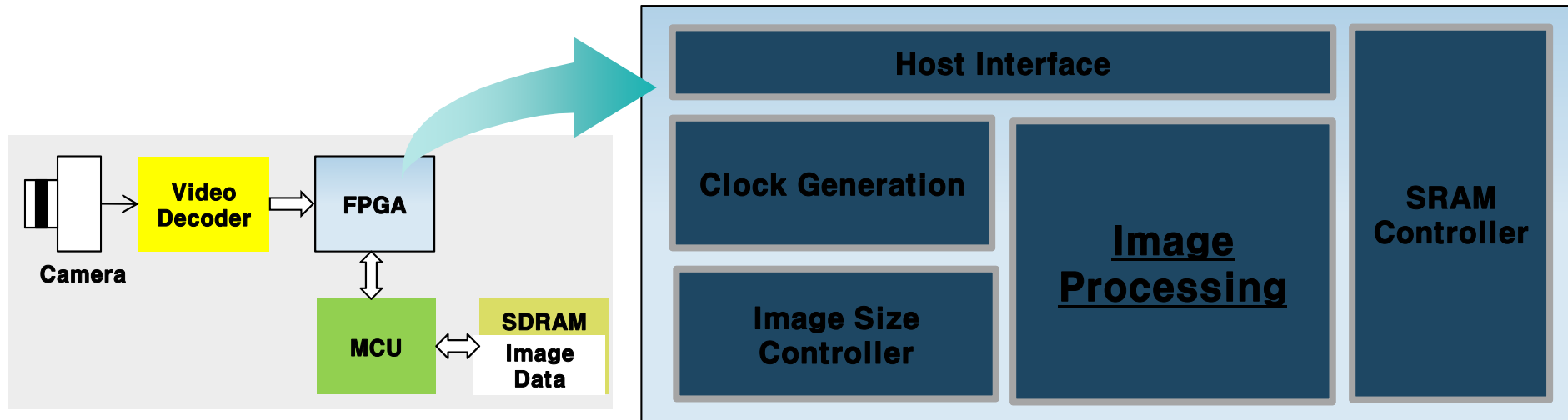




Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



FPGA Design Example



- **Clock Generation** : Video Decoder의 clock에 맞춰 영상처리를 위한 clock 생성
- **Host Interface** : MCU와 FPGA간의 데이터 전달 제어
- **SRAM Controller** : 처리된 영상 SRAM에 Read, Write
- **Image Size Controller** : 영상 축소 (720x480 → 180x120)
- **Image Processing** : Labeling, Noise 제거, Edge 검출 등
➡ (적, 아군, 장애물 인식 , 거리 및 위치 판단)

SDIA

Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!

ROBOTWAR

KIRIA 한국로봇산업진흥원
KOREA INSTITUTE FOR ROBOT INDUSTRY ADVANCEMENT

adc

ALTERA

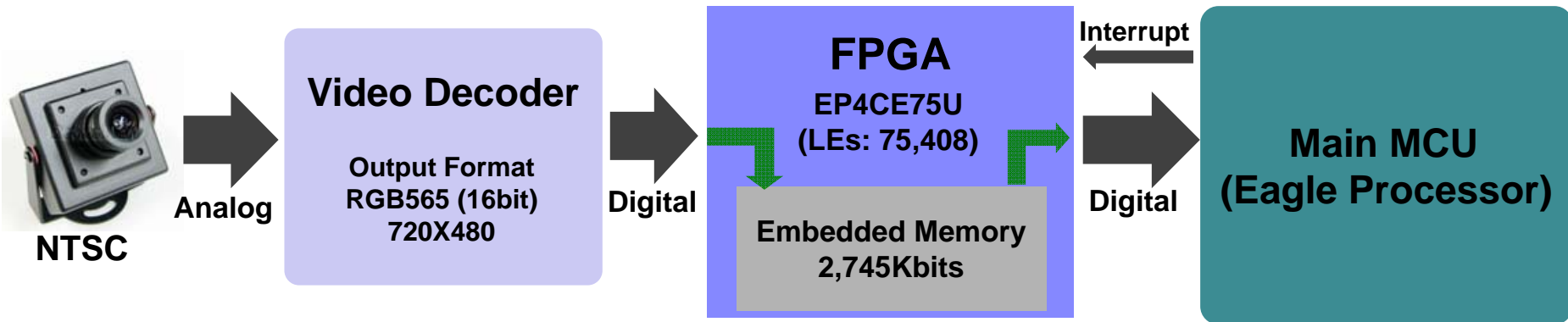
MINI ROBOT (주)미니로봇

ROBOTIS

IDC 반도체설계교육센터
IC DESIGN EDUCATION CENTER



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



Camera

- CCD Color Camera
- 30 frame/sec

Video Decoder

- Analog 영상을 디코딩
- 720X480, RGB565 출력
- 30 frame/sec

FPGA

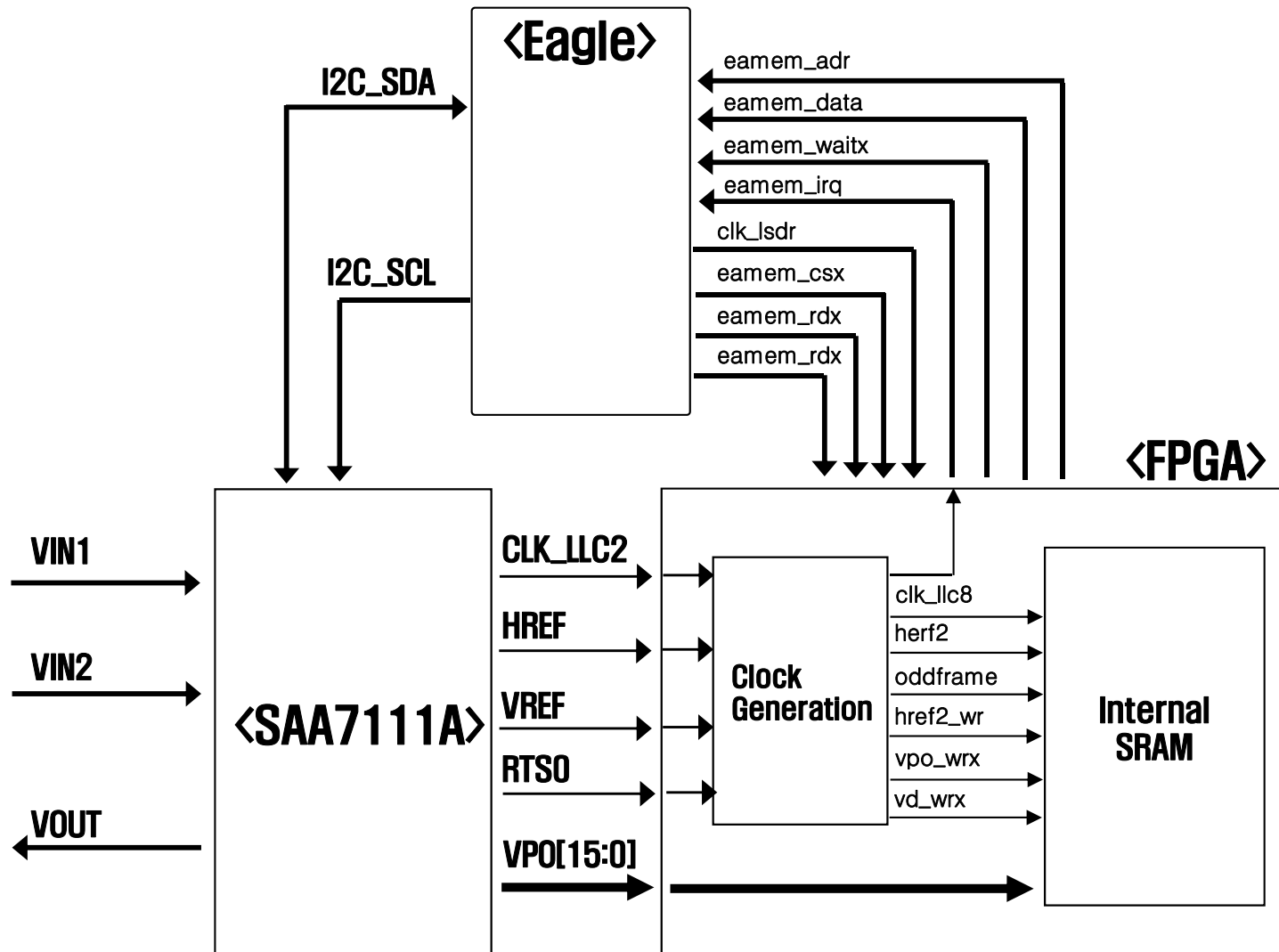
- Video Decoder로부터 Image Data 입력
- 영상 처리/인식
- Main MCU 요청에 의한 Image 전달

Main MCU

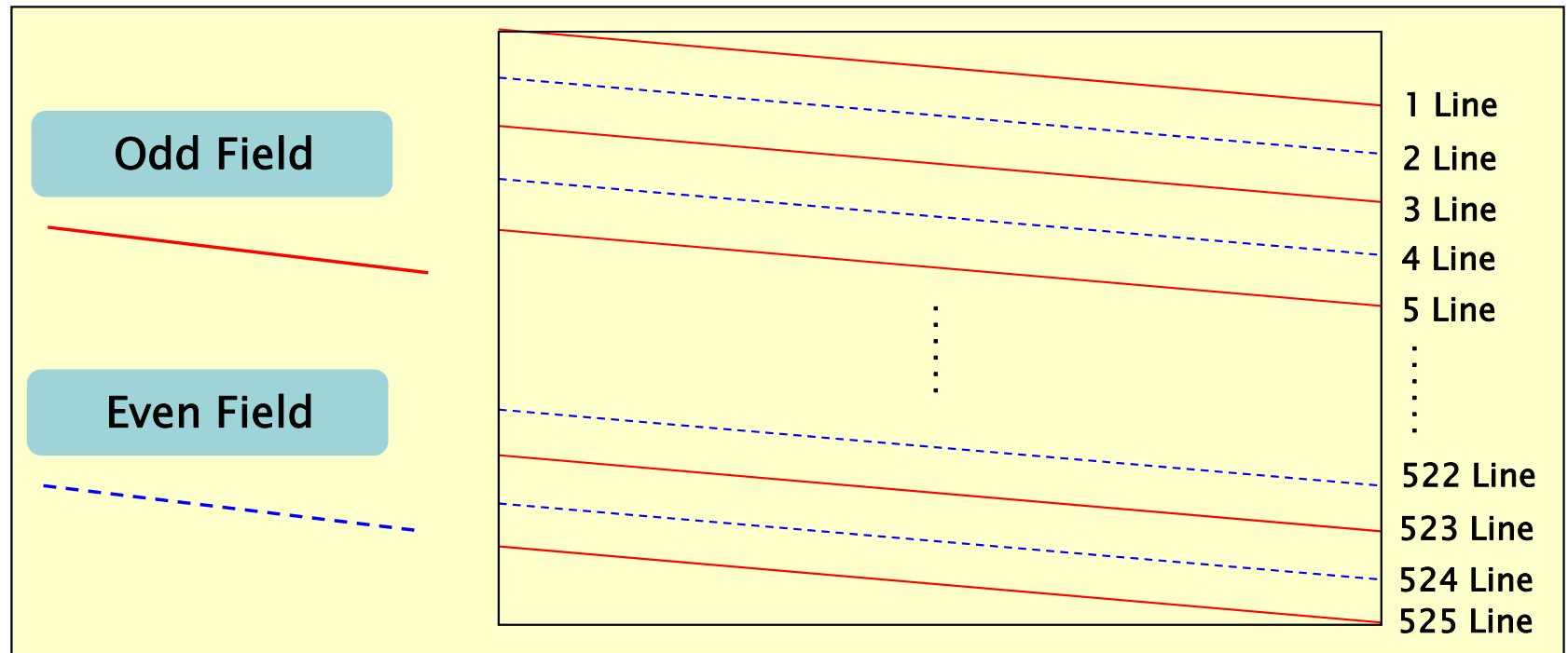
- Robot Strategy
- Decision
- Robot Motion SW



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



Scanning - NTSC



- NTSC (National Television System Committee)
- NTSC의 영상 Image는 Frame 당 525개의 수평 주사선을 가짐
- 왼쪽에서 오른쪽으로 그리고 위쪽에서 아래쪽으로 주사
- 한번은 홀수 번째줄(Odd Field), 다른 한번은 짝수 번째 줄(Even field)을 주사
- 절반의 Frame을 주사하는데 걸리는 시간은 1/60초(60Hz)
- 완전한 한 Frame은 1/30 초마다 주사

SDIA

Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!

ROBOTWAR

KIRIA 한국로봇산업진흥원
KOREA INSTITUTE FOR ROBOT INDUSTRY ADVANCEMENT

adc

ALTERA

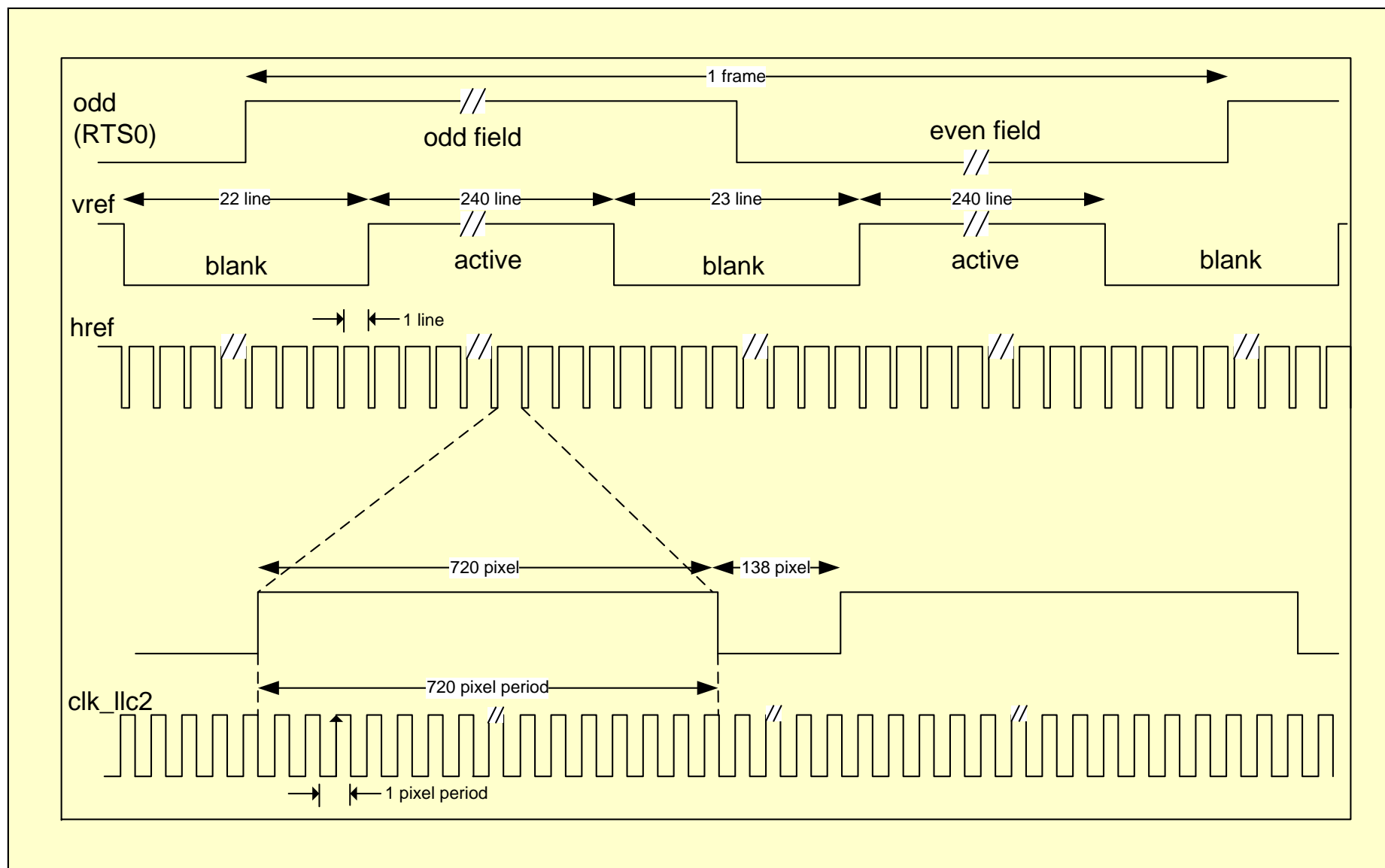
MINI (주)미니로봇 **ROBOT**

ROBOTIS

IDeC 반도체설계교육센터
IC DESIGN EDUCATION CENTER



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!





Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



◆ Input Signals of FPGA

- RTS0 : odd field와 even field를 구분해 주는 신호
→ odd field와 even field를 합했을 때 한 Frame이 완성
- VREF(Vertical Reference) : odd field와 even field에서 화면 위/아래의 blank 부분을 제외한 실제 필요한 영상 신호(active)를 구분하기 위한 신호
→ active 신호는 odd/even 각 Field에서 240개의 line으로 구성
- HREF(Horizontal Reference) : 한 line을 구분해 주는 신호
→ 1 Frame당 525번의 주기로써 변화
→ HREF의 한 주기(1 line) 동안 858개의 Image Pixel이 출력되는데, 실제 사용되는 Active Sample은 720개로 HREF 신호의 High구간에 해당
→ 각 Pixel은 CLK_LLC2 클럭의 Positive Edge에서 출력
- CLK_LLC2 : Video Decoding을 위한 System Clock(27MHz)의 $\frac{1}{2}$ Clock(13.5MHz)



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
// File Name: video_decoder.c
void SAA7111Initial()
{
    volatile U8 i, temp;

    // RGB(565) Setting
    U8 SAA7111_Buff[0x18] = {
        0x00, //sub_addr00 : chip version
        0x00, //sub_addr01 : not used
        0xc0, //sub_addr02
        0x23, //sub_addr03
        0x00, //sub_addr04
        0x00, //sub_addr05
        0xde, //sub_addr06
        0xdc, //sub_addr07
        0x40, //sub_addr08
        0x01, //sub_addr09
        0x80, //sub_addr0a
        0x47, //sub_addr0b
        0x40, //sub_addr0c
        0x00, //sub_addr0d
        0x01, //sub_addr0e
        0x00, //sub_addr0f
        0x00, //sub_addr10
        0x0c, //sub_addr11
        0x01, //sub_addr12
        0x00, //sub_addr13
        0x00, //sub_addr14
        0x00, //sub_addr15
        0x00, //sub_addr16
        0x00 //sub_addr17
    };
}
```

- MCU(Eagle)에서 SAA7111A Chip을 초기화 하여 FPGA에 입력되는 기본 신호 발생에 대한 설정

→ Analog Input signal Select

→ RGB(16bit), RGB(24bit), YUV(4:2:2)

- Eagle Chip의 I2C Control을 통해 SAA7111A Chip의 Sub Address를 Setting하여 영상 출력에 대한 신호 형태를 결정

(Sub Address Setting은 SAA7111A Datasheet 참고)

In/Out Signal Define

```
module FPGA_Processing ( resetx,  
  
    clk_llc2, clk_llc, vref, href, odd, vpo, // mem_ctrl <- SAA7111A  
  
    clk_lsdr, eamem_adr, eamem_data, eamem_csx, // Eagle Interface  
  
    eamem_wrx, eamem_rdx, eamem_waitx, eamem_irq, // Eagle Interface  
  
    led_test // FPGA test(LED On/Off)  
  
);  
  
input    resetx;  
  
/* mem_ctrl <- SAA7111A */  
input    clk_llc2; // 13.5 MHz  
input    clk_llc; // 27 MHz  
input    vref; // vertical sync.  
input    href; // horizontal sync.  
input    odd; // odd field (RTS0)  
input [15:0] vpo; // RGB(565) input vidoe data  
  
/* Eagle Interface */  
input    clk_lsdr; // sdram clk, Eagle SDRCLK  
input [17:0] eamem_adr; // Eagle Address[18:1]  
inout [15:0] eamem_data; // Eagle Data[15:0]  
input    eamem_csx; // FPGA Chip Select, Eagle nCS3  
input    eamem_wrx; // write strobe, Eagle nWR  
input    eamem_rdx; // read strobe, Eagle nRD  
output    eamem_waitx; // Eagle read wait, Eagle nWAIT  
output    eamem_irq; // external read interrupt(FPGA -> Eagle), Eagle IRQ6  
  
/* FPGA test */  
output    led_test;
```



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!





Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



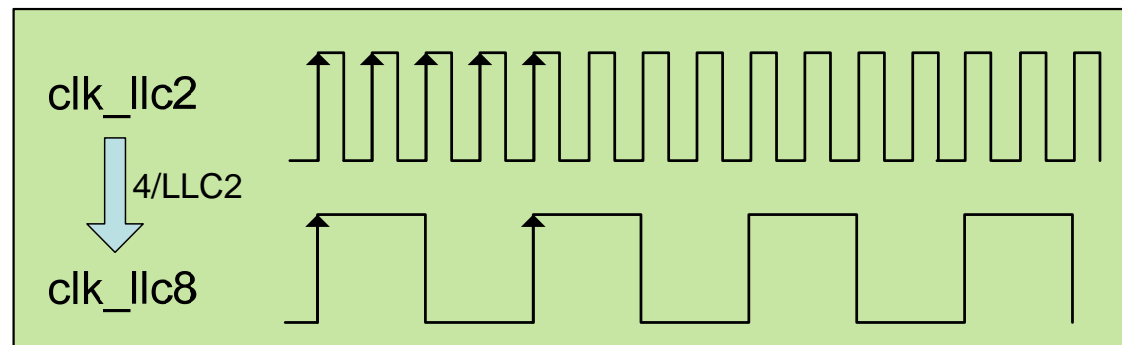
```
//-----
// Internal SRAM WRITE & Interrupt
// SAA7111A Video Decoder => SRAM, V/H sync. input
// 720x480 -> 180x120 compression
//-----

reg [ 1:0] clk_div;

always @(negedge resetx or posedge clk_llc2)
  if (~resetx)   clk_div <= 2'b0;
  else          clk_div <= clk_div + 1'b1;

// clk_llc8 : 180(720/4) clock generation
wire clk_llc8 = clk_div[1]; //-----(1)
```

(1) clk_llc8은 영상 신호의 한 Line의 720pixel 중 $\frac{1}{4}$ 만을 취하기 위한 클럭 소스로 아래의 그림과 같이 clk_llc2를 4분주하여 생성된다.



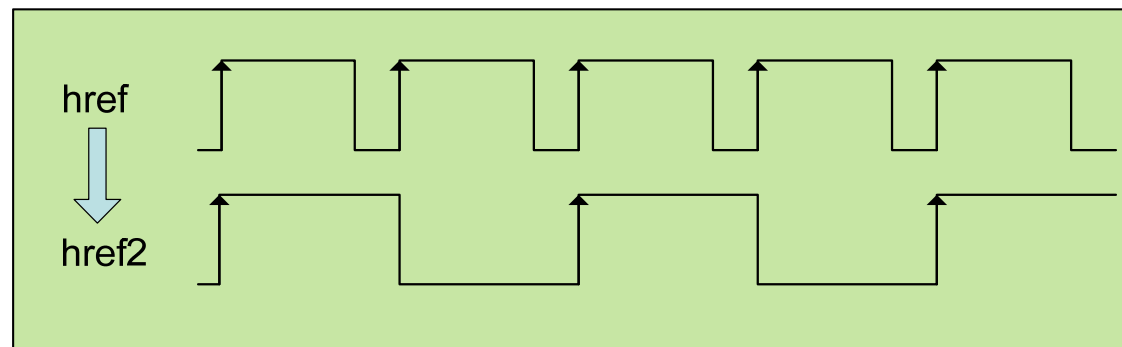


Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
// href2 : (480/2) clock generation
reg href2;
always @(negedge resetx or posedge href)
  if (~resetx) href2 <= 1'b0;
  else href2 <= ~href2; //------(2)
```

(2) href2는 active 영상 신호 480line 중 $\frac{1}{4}$ 만을 취하기 위한 과정에서 사용되는 신호로 (2)에서 생성되고, 아래 그림과 같이 href로부터 생성된다.



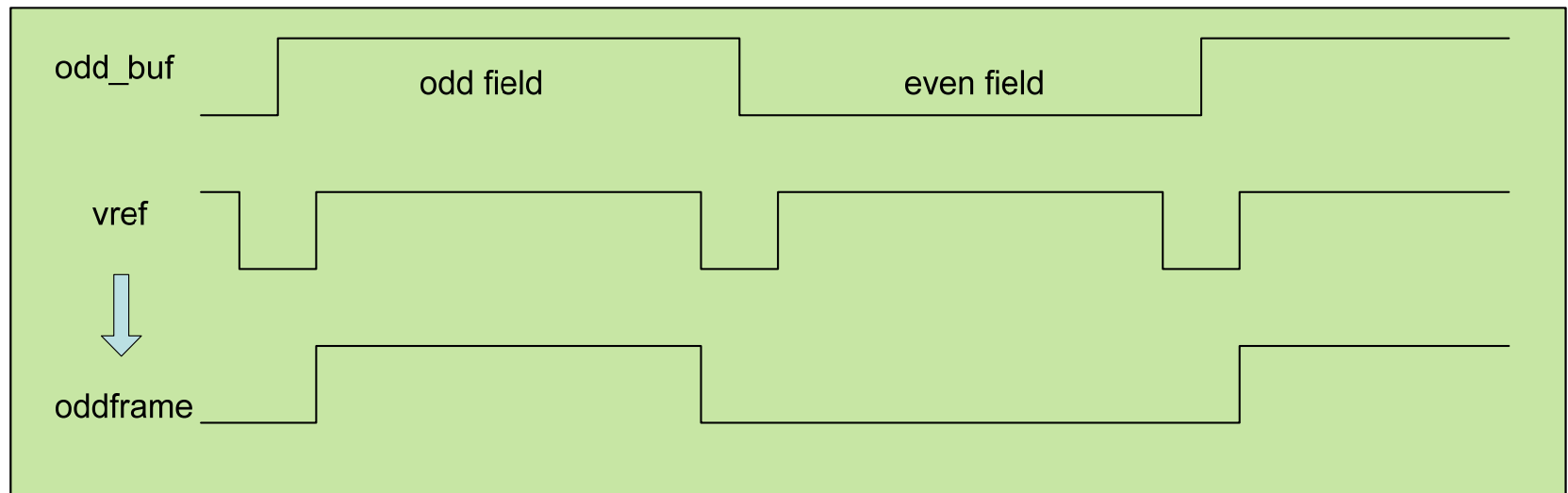


Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
// select only odd frame  
wire oddframe = odd & vref; //------(3)
```

(3) Oddframe은 (3)과 같으므로 vref 신호 중 odd field에 해당하는 active 구간 (240line)이 되고, 아래 그림과 같이 생성된다.





Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
// 120(480/4) clock generation
wire href2_wr = href2 & href & oddframe; //------(4)
// 180x120 write clock generation
wire vpo_wrx = ~(vref & href2_wr & clk_llc8); //------(5)

reg vpo_wrx1;
reg vpo_wrx2;
reg vpo_wrx3;

always @(negedge resetx or posedge clk_llc)
  if (~resetx) vpo_wrx1 <= 1'b1;
  else vpo_wrx1 <= vpo_wrx;
always @(negedge resetx or posedge clk_llc)
  if (~resetx) vpo_wrx2 <= 1'b1;
  else vpo_wrx2 <= vpo_wrx1;
always @(negedge resetx or posedge clk_llc)
  if (~resetx) vpo_wrx3 <= 1'b1;
  else vpo_wrx3 <= vpo_wrx2;
```

(4)에서는 href2와 href 신호를 AND 연산하여 line 수를 반으로 줄이고, oddframe 과 AND하여 다시 반으로 line 수를 줄인다. 결국 href2_wr은 active 480line 중 120line, 즉 $\frac{1}{4}$ 만을 취하는 형태의 신호이다.

(5)의 vpo_wrx는 120line 각각에 대해 clk_llc8과 AND 연산을 하는 형태이므로 각 line의 720pixel만을 취하는 형태의 신호가 된다. 그러므로 vpo_wrx는 720X480의 영상 중 가로/세로 각각 $\frac{1}{4}$ 만을 취해 180X120 크개의 영상을 얻기 위한 신호로 원래 영상을 $\frac{1}{16}$ 로 줄인 효과를 줄 수 있다.



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```

/// delayed write clock for no grich
wire vd_wrx = ~(~vpo_wrx1 & vpo_wrx3); //------(6)

//-----
// 16bit SRAM address generation (64KB)
// 180 x 120
//
// | | 0x0000
// | 180x120 |
// | |
// | |
// |-----| 0x5460
// | reserved |
// |-----| 0x8000
// | |
// | 180x120 |
// | |
// | |
// |-----| 0xD460
// | reserved |
// | | 0xFFFF
//
//-----
reg [15:0] vdata;
reg [17:0] vadr;
always @(negedge resetx or posedge clk_llc8)
    if (~resetx) vdata <= 16'b0;
    else if (href2_wr) vdata <= vpo; //------(7)
always @(negedge resetx or posedge clk_llc8)
    if (~resetx) vadr[14:0] <= 15'b0;
    else if (~oddframe) vadr[14:0] <= 15'b0;
    else if (href2_wr) vadr[14:0] <= vadr[14:0] + 1'b1; //(8)
always @(negedge resetx or posedge odd)
    if (~resetx) vadr[17:15] <= 3'b0;
    else vadr[15] <= ~vadr[15]; //------(9)
    
```

Vpo_wrx는 (9)에서 vd_wrx신호를 생성하여 SRAM에 영상 데이터를 write할때 사용된다.

(7)에서는 실제 사용되는 180X120의 영상 데이터를 vdata[15:0] Register에 가져오게 된다.

(8)과 (9)에서 데이터가 저장될 SRAM의 Address가 결정된다.

(9)를 통해 SRAM의 저장 영역을 2개로 나누어 번갈아 가면서 데이터를 저장할 수 있다.



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
//-----  
// External Interrupt Generation  
// 1 interrupter per 1 frame(interrupt length = clk_lsdr 2cycle)  
//-----  
reg oddframe_d1;  
reg oddframe_d2;  
reg oddframe_d3;  
always @(negedge resetx or posedge clk_lsdr)  
    if (~resetx)        oddframe_d1 <= 1'b0;  
    else                 oddframe_d1 <= oddframe;  
always @(negedge resetx or posedge clk_lsdr)  
    if (~resetx)        oddframe_d2 <= 1'b0;  
    else                 oddframe_d2 <= oddframe_d1;  
always @(negedge resetx or posedge clk_lsdr)  
    if (~resetx)        oddframe_d3 <= 1'b0;  
    else                 oddframe_d3 <= oddframe_d2;  
  
assign eamem_irq = ~oddframe_d1 & oddframe_d3; //------(10)
```

(10)은 oddframe 신호를 통해 eamem_irq 신호를 생성한 것으로 한 Frame이 처리될 때 마다 인터럽트를 발생시키기 위한 신호이다.



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!

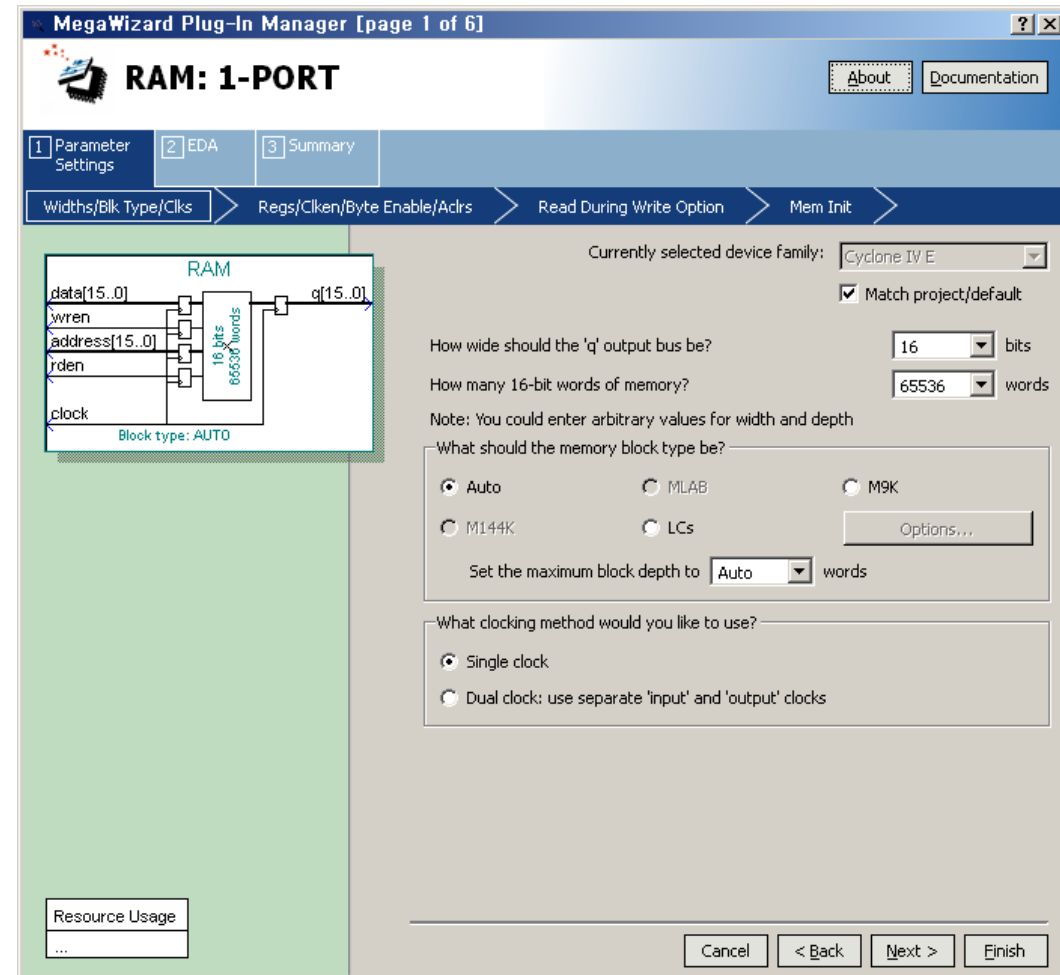


```

wire      [15:0] vmem_addr;
wire      [15:0] vmem_data;
wire      vmem_rden;
wire      vmem_wren;
wire [15:0] vmem_q;
    
```

```

RAM      RAM_inst (
    .address ( vmem_addr ),
    .clock ( clk_lsdr ),
    .data ( vmem_data ),
    .rden ( vmem_rden ),
    .wren ( vmem_wren ),
    .q ( vmem_q )
);
    
```





Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
reg [6:0] cs, ns;
```

```
parameter s0 = 7'b0000001;  
parameter s1 = 7'b0000010;  
parameter s2 = 7'b0000100;  
parameter s3 = 7'b0001000;  
parameter s4 = 7'b0010000;  
parameter s5 = 7'b0100000;  
parameter s6 = 7'b1000000;
```

```
wire mcs0 = cs[0]; // idle state  
wire mcs1 = cs[1]; // sa7111 video data write state  
wire mcs2 = cs[2]; // sa7111 video data write last state  
wire mcs3 = cs[3]; // Eagle data write state(for test)  
wire mcs4 = cs[4]; // Eagle data write last state  
wire mcs5 = cs[5]; // Eagle data read state  
wire mcs6 = cs[6]; // Eagle data read last state
```

```
always @(negedge resetx or posedge clk_lsdr)  
  if (~resetx) cs <= s0;  
  else      cs <= ns;
```



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
always @(mcs0 or mcs1 or mcs2 or mcs3 or mcs4 or mcs5 or mcs6 or eamem_csx or
vd_wrx or eamem_wrx or eamem_rdx) begin
  ns = s0;
  case (vdd) // synopsys parallel_case full_case
    mcs0 : if ( ~vd_wrx )          ns = s1;
           else if ( vd_wrx & ~eamem_csx & ~eamem_wrx ) ns = s3;
           else if ( vd_wrx & ~eamem_csx & ~eamem_rdx ) ns = s5;
           else                    ns = s0;
    mcs1 : if (vd_wrx)             ns = s2;
           else                    ns = s1;

    mcs2 :                          ns = s0;

    mcs3 : if (eamem_wrx )          ns = s4;
           else                    ns = s3;

    mcs4 :                          ns = s0;

           mcs5 : if (eamem_rdx)    ns = s6;
           else                    ns = s5;

    mcs6 :                          ns = s0;

    default :                      ns = s0;
  endcase
end
```



Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



```
//-----  
// SRAM Controller Output  
//-----  
//assign mem_csx    = mcs0;                // SRAM Chip select  
  
assign vmem_wren    = mcs1;                // SRAM Write // ~( mcs1 );  
  
assign vmem_rden    = mcs5;                // SRAM Read //~mcs5;  
  
assign eamem_data   = ( ~eamem_csx ) ? vmem_q : 16'bZ; //---(11)  
  
assign vmem_data    = ( mcs1 | mcs2 ) ? vdata : 16'bZ ; //---(12)  
  
assign vmem_addr    = ( mcs1 | mcs2 ) ? vadr : eamem_adr; //---(13)
```

(11)에서 (13)까지는 SRAM Controller에 대한 설계 부분으로 상태 신호에 대한 정의와 데이터 Read/Write, Address 할당 등 전반적인 SRAM 제어에 관한 부분이다.

(11)에서 eagle MCU로 영상 데이터를 넘겨주게 되고,

(12)에서 SRAM으로 영상 데이터를 저장하게 된다.

(13)은 처리되는 데이터의 Address 할당 부분이다.

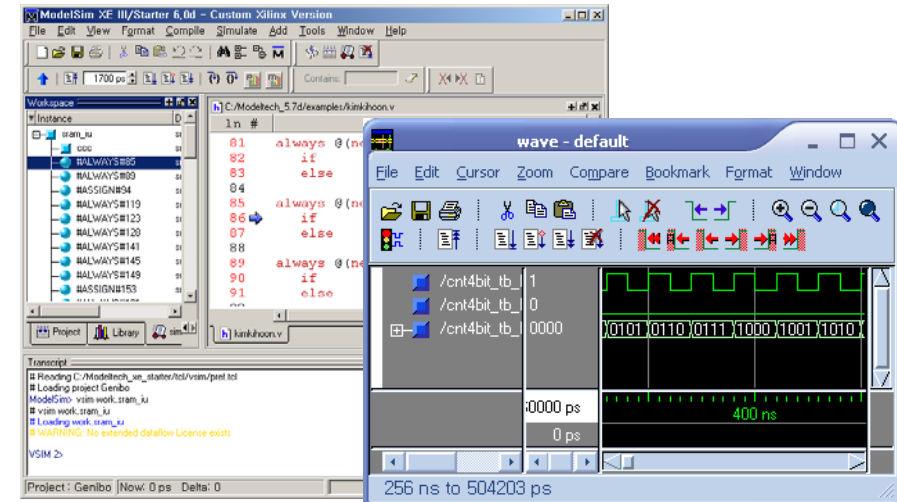


Excellence in
Intelligent Robot,
Wearable Computer,
and Bio/Health!



ModelSim (Simulation)

- VHDL, Verilog, SystemC
- Simulation
- Website: www.modelsim.com



Quartus II (Compilation)

- Synthesis, Implementation
- FPGA Download
- Website: www.altera.com

