//Includes

**#include** "altera\_avalon\_timer\_regs.h"

**#include** "altera\_avalon\_pio\_regs.h"

**#include** <io.h>

**#include** "altera\_avalon\_pwm\_regs.h"

**#include** "altera\_avalon\_pwm\_routines.h"

**#include** "system.h"

**#include** <stdio.h>

**#include** "sys/alt\_alarm.h"

**#include** "sys/alt\_timestamp.h"

**#include** "alt\_types.h"

**#include** <unistd.h>

**#include** <math.h>

**#include** "sys/alt\_irq.h"

**#include** "chu\_avalon\_gpio.h"

**#include** "chu\_avalon\_vga.h"

**#include** <stdlib.h>

//Function Protypes

**void** **print\_error**(**unsigned** **int** address, **int** return\_code);

**void** **check\_return\_code**(**unsigned** **int** address, **int** return\_code);

**int** initRPM=0;

**char** IRPM[4],SRPM[4];

**int** set\_rpm=3100; // MAX 3350

**int** srpm;

**int** i=100;

**float** xpl =40.0;

**float** ypl, step;

**int** prx=0,pry=300;

**void** **plot\_RPM**(alt\_u32 vga\_base,**int** rpm,**int** duty){

**const** **float** XMAX=5.0; // max range of x-axis

**const** **float** YMAX=5.0; // max range of y-axis

**int** j;

step = XMAX / (**float**)(DISP\_GRF\_X\_MAX);

xpl = xpl + step;

ypl = 0.001 \* rpm;

**if**(xpl>600){

**int** x=40,y=80;

vga\_clr\_screen(VRAM\_BASE,0xff);

**for**(x=0; x<40; x++)

**for**(y=0; y<480; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B);

**for**(x=600; x<640; x++)

**for**(y=0; y<480; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B);

**for**(x=0; x<640; x++)

**for**(y=0; y<80; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B);

**for**(x=0; x<640; x++)

**for**(y=400; y<480; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B);

x=40,y=80;

**while**(y<=400){

vga\_plot\_line(VRAM\_BASE,x,y,DISP\_GRF\_X\_MAX-40,y,0x1c);

y+=10;

}

x=40,y=80;

**while**(x<=600){

vga\_plot\_line(VRAM\_BASE,x,y,x,DISP\_GRF\_Y\_MAX-80,0x1c);

x+=10;

}

x=40,y=45;

vga\_plot\_line(VRAM\_BASE,x,y,DISP\_GRF\_X\_MAX-40,y,0x44);

vga\_plot\_line(VRAM\_BASE,x,45,x,DISP\_GRF\_Y\_MAX-40,0x44);

vga\_plot\_line(VRAM\_BASE,DISP\_GRF\_X\_MAX-40,45,DISP\_GRF\_X\_MAX-40,DISP\_GRF\_Y\_MAX-40,0x44);

vga\_plot\_line(VRAM\_BASE,x,DISP\_GRF\_Y\_MAX-40,DISP\_GRF\_X\_MAX-40,DISP\_GRF\_Y\_MAX-40,0x44);

**sprintf**(SRPM,"%i",set\_rpm);

vga\_wr\_bit\_str(VRAM\_BASE,280,55,"SET RPM:",0x44,1);

vga\_wr\_bit\_str(VRAM\_BASE,345,55,SRPM,0x03,1);

vga\_wr\_bit\_str(VRAM\_BASE,140,410,"Time elapsed:",0x44,1);

vga\_wr\_bit\_str(VRAM\_BASE,360,410,"Duty Cycle :",0x44,1);

vga\_wr\_bit\_str(VRAM\_BASE,45,410,"Status:",0x22,1);

vga\_wr\_bit\_str(VRAM\_BASE,100,55,"Initial RPM:",0x44,1);

**sprintf**(IRPM,"%i",initRPM);

vga\_wr\_bit\_str(VRAM\_BASE,200,55,IRPM,0x03,1);

vga\_wr\_bit\_str(VRAM\_BASE,20,20,"FPGA Controls Systems",0x00,1);

vga\_wr\_bit\_str(VRAM\_BASE,400,55,"RPM:",0x44,1);

xpl=40.0;

}

**if** (ypl < YMAX){ // plot if only y is in range

j = DISP\_GRF\_Y\_MAX-(ypl/YMAX)\*DISP\_GRF\_Y\_MAX;

vga\_wr\_pix(vga\_base, xpl++, j, 0x00);

}

}

**int** numRevsMotor=0;

**volatile** **int** edge\_capture;

**void** **write\_pixel**(**int** x, **int** y, **short** colour) {

**volatile** **short** \*vga\_addr=(**volatile** **short**\*)(0x08000000 + (y<<10) + (x<<1));

\*vga\_addr=colour;

}

alt\_u8 **sseg\_32conv\_hex**(**int** hex)

{

/\* active-low hex digit 7-seg patterns (0-9,a-f); MSB ignored \*/

**static** **const** alt\_u8 SSEG\_HEX\_TABLE[16] = {

0x40, 0x79, 0x24, 0x30, 0x19, 0x92, 0x02, 0x78, 0x00, 0x10, //0-9

0x88, 0x03, 0x46, 0x21, 0x06, 0x0E}; //a-f

alt\_u8 ptn;

**if** (hex < 16)

ptn = SSEG\_HEX\_TABLE[hex];

**else**

ptn = 0xff;

**return** (ptn);

}

**int** **main**(**void**){

**char** \*strduty;

vga\_clr\_screen(VRAM\_BASE,0xff);

**volatile** **int** duty\_cycle;

**int** return\_code = ALTERA\_AVALON\_PWM\_OK;

//Initialize PWM and Check Return Code

return\_code = altera\_avalon\_pwm\_init(Z\_PWM\_0\_BASE, 500000, 1);

check\_return\_code(Z\_PWM\_0\_BASE, return\_code);

//Enable PWM and Check Return Code

return\_code = altera\_avalon\_pwm\_enable(Z\_PWM\_0\_BASE);

check\_return\_code(Z\_PWM\_0\_BASE, return\_code);

//init duty\_cycle with the value written to duty\_cycle register during initialization

duty\_cycle = IORD\_ALTERA\_AVALON\_PWM\_DUTY\_CYCLE(Z\_PWM\_0\_BASE);

altera\_avalon\_pwm\_disable(Z\_PWM\_1\_BASE);

IOWR\_ALTERA\_AVALON\_PIO\_DATA(HBRIDGE\_ENABLE\_BASE,1);

duty\_cycle = 10; // ( 1 - 100000 )

return\_code = altera\_avalon\_pwm\_change\_duty\_cycle(Z\_PWM\_0\_BASE, duty\_cycle);

check\_return\_code(Z\_PWM\_0\_BASE, return\_code);

**int** first\_val, second\_val;

**int** intit\_pulse\_val,final\_pulse\_val;

**int** RPM;

srpm=set\_rpm;

**float** Error\_term=0, P\_Term=0, I\_Term=0, D\_Term=0, D\_State=0, I\_Gain=5, P\_Gain=15, D\_Gain=2;

**int** x,y;

// --------- Σχεδιασμός ορθογωνίων ---------- //

**for**(x=0; x<40; x++)

**for**(y=0; y<480; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B); //3b

**for**(x=600; x<640; x++)

**for**(y=0; y<480; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B);

**for**(x=0; x<640; x++)

**for**(y=0; y<80; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B);

**for**(x=0; x<640; x++)

**for**(y=400; y<480; y++)

vga\_wr\_pix(VRAM\_BASE, x, y, 0x3B); //DA ,r 0x03

// ----------- Επικεφαλίδες ---------------//

vga\_wr\_bit\_str(VRAM\_BASE,20,20,"ZNUC II - CYCLONE II PID MOTOR CONTROL SYSTEM",0x00,1);

vga\_wr\_bit\_str(VRAM\_BASE,400,55,"RPM:",0x44,1);

// ----------- Σχεδιασμός πλεγματος ------//

x=40,y=80;

**while**(y<=400){

vga\_plot\_line(VRAM\_BASE,x,y,DISP\_GRF\_X\_MAX-40,y,0x1c);

y+=10;

}

x=40,y=80;

**while**(x<=600){

vga\_plot\_line(VRAM\_BASE,x,y,x,DISP\_GRF\_Y\_MAX-80,0x1c);

x+=10;

}

// --------------- Περίγραμμα πλαισίου ----------------//

x=40,y=45;

vga\_plot\_line(VRAM\_BASE,x,y,DISP\_GRF\_X\_MAX-40,y,0x44);

vga\_plot\_line(VRAM\_BASE,x,45,x,DISP\_GRF\_Y\_MAX-40,0x44);

vga\_plot\_line(VRAM\_BASE,DISP\_GRF\_X\_MAX-40,45,DISP\_GRF\_X\_MAX-40,DISP\_GRF\_Y\_MAX-40,0x44);

vga\_plot\_line(VRAM\_BASE,x,DISP\_GRF\_Y\_MAX-40,DISP\_GRF\_X\_MAX-40,DISP\_GRF\_Y\_MAX-40,0x44);

// ----------------- Επικεφαλιδες ---------------------//

vga\_wr\_bit\_str(VRAM\_BASE,140,410,"Time elapsed:",0x44,1);

vga\_wr\_bit\_str(VRAM\_BASE,360,410,"Duty Cycle :",0x44,1);

vga\_wr\_bit\_str(VRAM\_BASE,45,410,"Status:",0x22,1);

vga\_wr\_bit\_str(VRAM\_BASE,100,55,"Initial RPM:",0x44,1);

**int** previous\_duty=0;

**while**(1){

second\_val = 0;

**sprintf**(SRPM,"%i",set\_rpm);

vga\_wr\_bit\_str(VRAM\_BASE,280,55,"SET RPM:",0x44,1);

vga\_wr\_bit\_str(VRAM\_BASE,345,55,SRPM,0x03,1);

// ------ RMP computation in 0.1 sec delay ---- //

first\_val = alt\_nticks();

intit\_pulse\_val = IORD\_32DIRECT(VELOCITY\_BASE,0);

**while**((second\_val-first\_val)<100){ // 0.1 sec

second\_val = alt\_nticks();

}

final\_pulse\_val = IORD\_32DIRECT(VELOCITY\_BASE,0);

numRevsMotor = (final\_pulse\_val-intit\_pulse\_val);

RPM = **abs**((numRevsMotor\*600)/1440);

// ------------------------------------------- //

plot\_RPM(VRAM\_BASE,RPM,RPM);

**int** xt, yt;

**for**(xt=460; xt<510; xt++)

**for**(yt=410; yt<425; yt++)

vga\_wr\_pix(VRAM\_BASE, xt, yt, 0x3B);

**for**(xt=440; xt<475; xt++)

**for**(yt=55; yt<75; yt++)

vga\_wr\_pix(VRAM\_BASE, xt, yt, 0x3B);

**for**(xt=265; xt<295; xt++)

**for**(yt=410; yt<425; yt++)

vga\_wr\_pix(VRAM\_BASE, xt, yt, 0x3B);

**if**(srpm!=set\_rpm)

{

**for**(xt=345; xt<380; xt++)

**for**(yt=55; yt<70; yt++)

vga\_wr\_pix(VRAM\_BASE, xt, yt, 0x3B);

set\_rpm=srpm;

}

**if**(initRPM==0){

**sprintf**(strduty,"%i",RPM);

vga\_wr\_bit\_str(VRAM\_BASE,200,55,strduty,0x03,1);

initRPM=RPM;

}

**sprintf**(strduty,"%i",RPM);

vga\_wr\_bit\_str(VRAM\_BASE,440,55,strduty,0x03,1);

// --------------- PID Computation ----------- //

Error\_term = (**float**)set\_rpm - RPM;

P\_Term = P\_Gain \* Error\_term;

D\_Term = D\_Gain \* (Error\_term - D\_State);

D\_State = Error\_term;

I\_Term = I\_Term + I\_Gain \* Error\_term;

duty\_cycle = duty\_cycle - (P\_Term + I\_Term + D\_Term);

// ------------------------------------------ //

return\_code = altera\_avalon\_pwm\_change\_duty\_cycle(Z\_PWM\_0\_BASE,duty\_cycle);

**if**(return\_code==-1) //duty cycle reg must be less than or equal to clock divider

{

duty\_cycle = previous\_duty;

return\_code = altera\_avalon\_pwm\_change\_duty\_cycle(Z\_PWM\_0\_BASE,duty\_cycle);

}

check\_return\_code(Z\_PWM\_0\_BASE, return\_code);

**sprintf**(strduty,"%u32",alt\_nticks()/alt\_ticks\_per\_second());

vga\_wr\_bit\_str(VRAM\_BASE,265,410,strduty,0x03,1);

vga\_wr\_bit\_str(VRAM\_BASE,300,410,"sec",0x44,1);

previous\_duty=duty\_cycle;

**if**(alt\_nticks()/alt\_ticks\_per\_second()==20){

srpm=3300;

}

**if**(alt\_nticks()/alt\_ticks\_per\_second()==40){

srpm=2900;

}

**if**(alt\_nticks()/alt\_ticks\_per\_second()>=60){

srpm=3400;

}

**if**(alt\_nticks()/alt\_ticks\_per\_second()>=80){

srpm=3200;

}

}

**return** 0;

}