Xilinx Zynq FPGA,TI DSP, MCU 기반의 프로그래밍 전문가 과정

날 짜 : 2018 . 5. 10

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 정한별
hanbulkr@gmail.com

Aduino motor SET

//rotation physics with arduino

```
#include <Servo.h>
#define DT 100.0      // delay time
#define DTHETA 20.0 //20 도 씩 돌아간다는 뜻이다. ->> 1/9 파이.

Servo myservo;

int theta =0;

double omega;   // 각속도.(rad/sec)인데 (도/sec)로 되어 있음 (2 파이/180 도)
double alpha;   // 각가속도.
double velocity;  // 속도.
double acceleration;  // 가속도.
double dt = DT /1000.0; // 0.1 초
double time = 0.0;


void setup() {

  Serial.begin(9600);             // 시리얼 보레이트 설정
  myservo.attach(9);

}

void loop() {

  Serial.println((double)(DTHETA/(500.0/1000.0)));  //40
  for(theta =0; theta <180; theta += DTHETA)      // 세타 값이 0 부터 180 되기전까지 20 도씩 올림.
  {
    myservo.write(theta);       // 서보모터를 사용// 1 회 0 // 2 회 20 // 3 회 40
    delay(DT);                  // 1 회 0.1 초 // 2 회 100 //
    Serial.print("Theta = ");   // 1 회 0
    Serial.println(theta);      // 1 회 0 // 2 회 20 // 3 회 40
    Serial.print("DTheta = "); // 1 회 0
    Serial.println(DTHETA);     // 1 회 0
    time += dt;                 // 1 회 0.1 초 //2 회 0.2 초 //3 회 0.3 초
    Serial.print("Total Time = ");        // 1 회 0.0
    Serial.print(time);                   // 1 회 0.0
    Serial.print(" dt= ");                // 1 회 0.1
    Serial.print(dt);                     // 1 회 0.1
    Serial.print(" Omega = "); // 1 회 200 //
    Serial.println((double)(DTHETA)/dt);        // 1 회 200 // 2 회 200 // 3 회 300
    Serial.println();      // 개행//
    //delay(1500);
  }
  for(theta = 180; theta >= 1; theta -= DTHETA)   // 세타 값이 1 이 되기까지 세타를 20 씩 감소함.
  {
    myservo.write(theta);
    delay(DT);
  }

}
```

//ramdom_number_control_with_arduino

```
int randNumber;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  Serial.println("Print Random Numbers 0~9");

  for(int i =0; i<20; i++)
  {
    randNumber = random(10);        // 0~9 까지의 랜덤값을 저장.
    Serial.print(randNumber); //
    Serial.print(" ");
  }

  Serial.println();
  Serial.println("Print Random Numbers 2~9");

  for(int i= 0; i<20; i++)
  {
    randNumber = random(2,10);      // 0~9 까지의 랜덤값을 저장.
    Serial.print(randNumber);
    Serial.print(" ");
  }
  randomSeed(analogRead(0));        // srand 해서 랜덤값이 겹치게 출력하지 않게 한다.
  Serial.println();
  Serial.println("Print Random Numbers 0~9");

  for(int i=0; i<20; i++)
  {
    randNumber = random(10);        // 0~9 까지의 랜덤 값을 출력한다.
    Serial.print(randNumber);
    Serial.print(" ");

  }
  Serial.println();
  Serial.println();

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

// 수치미분

```
#include<Servo.h>

#define DT 50.0

Servo myservo;
// 정확도를. 위해 길게. 쓴다.
double pi = 3.14159265358979323846264338327950288419716939937510592O9;

int theta = 0;            // 각도 의변화
double d_theta= 0.0;

double radian = 0.0;            // 180/파이;
double omega =0.0;
double d_omega = 0.0;        // 현재. 측정한. 오메가.-이전오메가.

double alpha = 0.0;
double velocity;
double acceleration;

double dt = DT /1000.0; // 0.05 초
double time = 0.0;


void setup() {

   Serial.begin(9600);
   randomSeed(analogRead(0));      // 랜덤으로 출력되는 값이 겹치지 않게 해줌.
   myservo.attach(9);           // 9 번핀을 사용.

}

void loop() {

  while(theta <180)
  {
   myservo.write(theta);
   delay(DT);

   Serial.print("Theta = ");
   Serial.println(theta);

   Serial.print("DTheta = ");
   Serial.println(d_theta);

   Serial.print("Radian = ");
   radian = (d_theta /360)*2*pi;
   Serial.println(radian);

   time += dt;

   Serial.print("Total Time = ");
   Serial.println(time);
   Serial.print("dt = ");
   Serial.println(dt);

   Serial.print("Omega = ");
   d_omega = (radian /dt) - omega;   //음수가 나올수있는데 torque 가바뀐다는 의미.
   omega = radian /dt;
   Serial.println(omega);
```

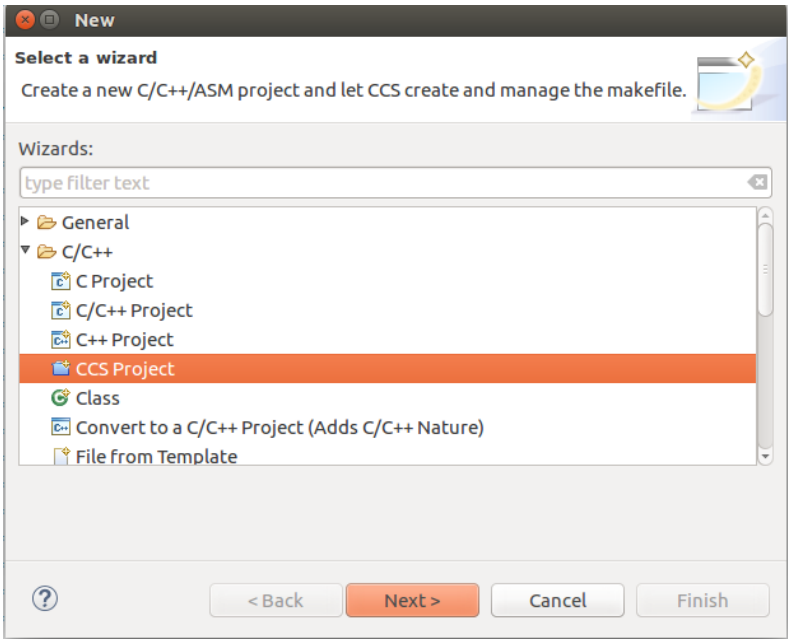
```
    Serial.print("DOmega = ");
    Serial.println(d_omega);

    Serial.print("Velocity = ");
    velocity = 0.01815 * omega ;
    Serial.println(velocity);

    Serial.print("Acceleration = ");
    acceleration = 0.01815 * omega *omega;
    Serial.println(acceleration, 10);   // 10 진수 로출력해라.

    Serial.print("Alpha = ");
    alpha = d_omega /dt;    // d_omega 가'-'일수있어서 alpha 도 -일수있다.
    Serial.println(alpha);

    Serial.println();   // 개행문자.

    d_theta = random(1, 11);
    theta +=d_theta;
  }
  for(theta =180; theta >= 1; theta -= random(1,11))
  {
    myservo.write(theta);
    delay(DT);
  }
}
```

CORETEX-R5 → Halcogen 이용하기 (기본 셋팅)

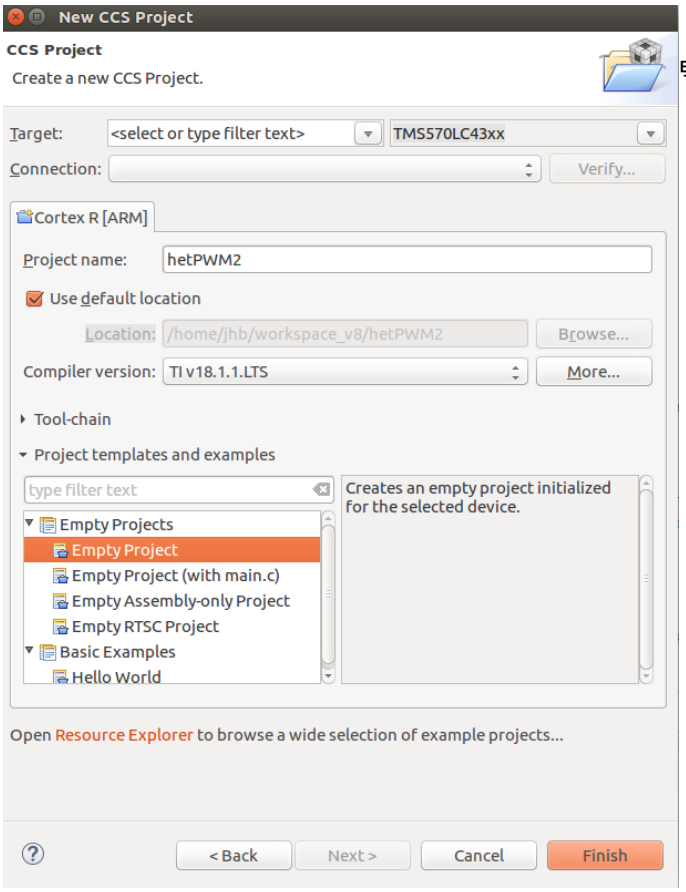
1. ccs 프로그램을 켠다.

2. project 를 만든다.

(1).



(2).



<1> TMS570LC43xx 선택

<2> 이름 설정.

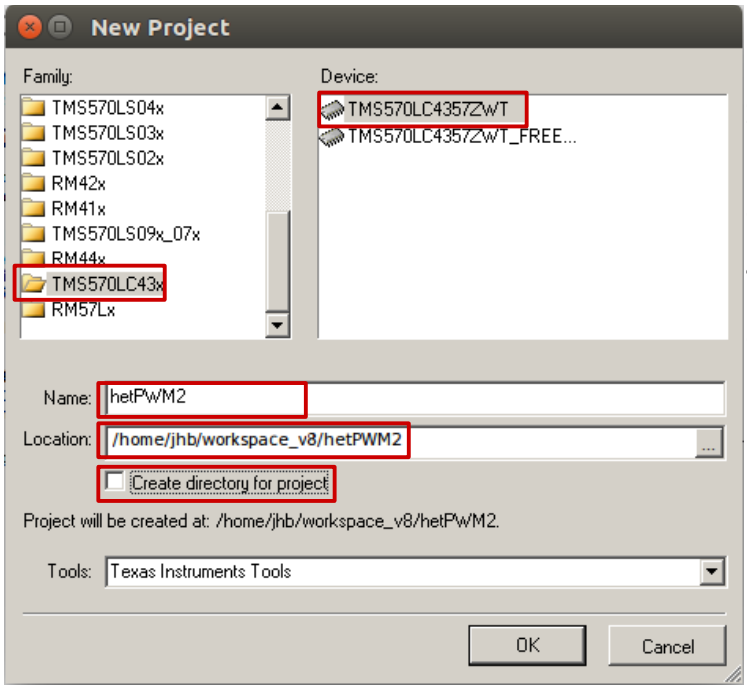
<3> Empty Project 설정.

3. 새로 만든창의 속성창을 켠다. (새로만든 프로젝트 클릭 후 Alt + Enter) 를 눌려서 볼 수 있다.

4. 경로의 위치를 파악한다. Resource 에 보면 경로가 있고 이 경로를 복사한다.

5. halcogen 을 킨다.

6. File → new → project 를 눌린다.



1)TMS570LC43x 선택한다.

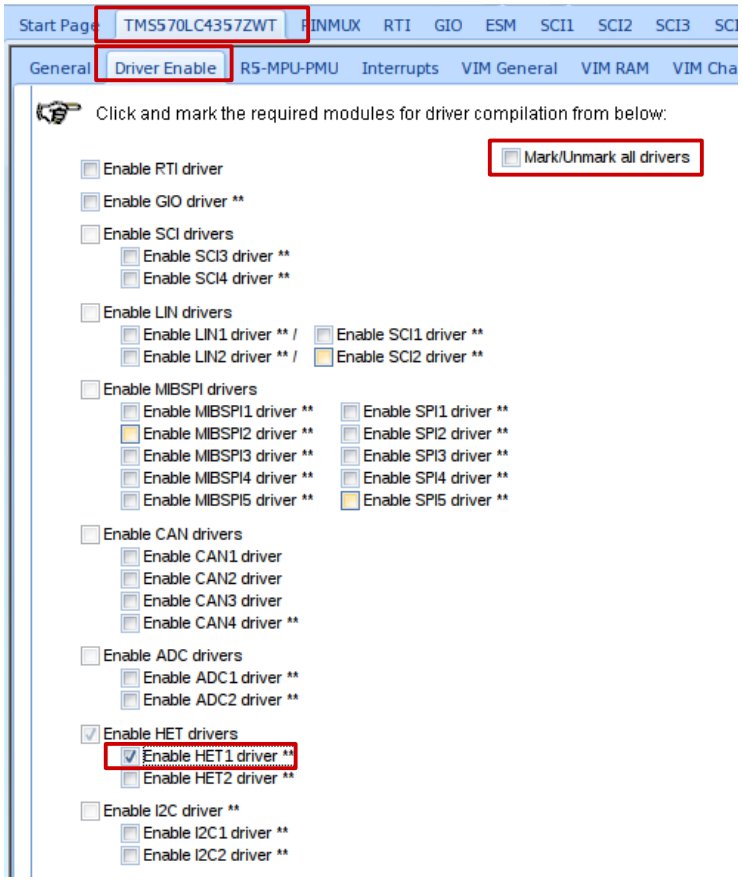
2) TMS5707ZWT 를 선택한다.

3) 이름을 정한다.

4) 좀전에 CCS 에서 복사한 경로 삽입

5) Create directroy 는 해제

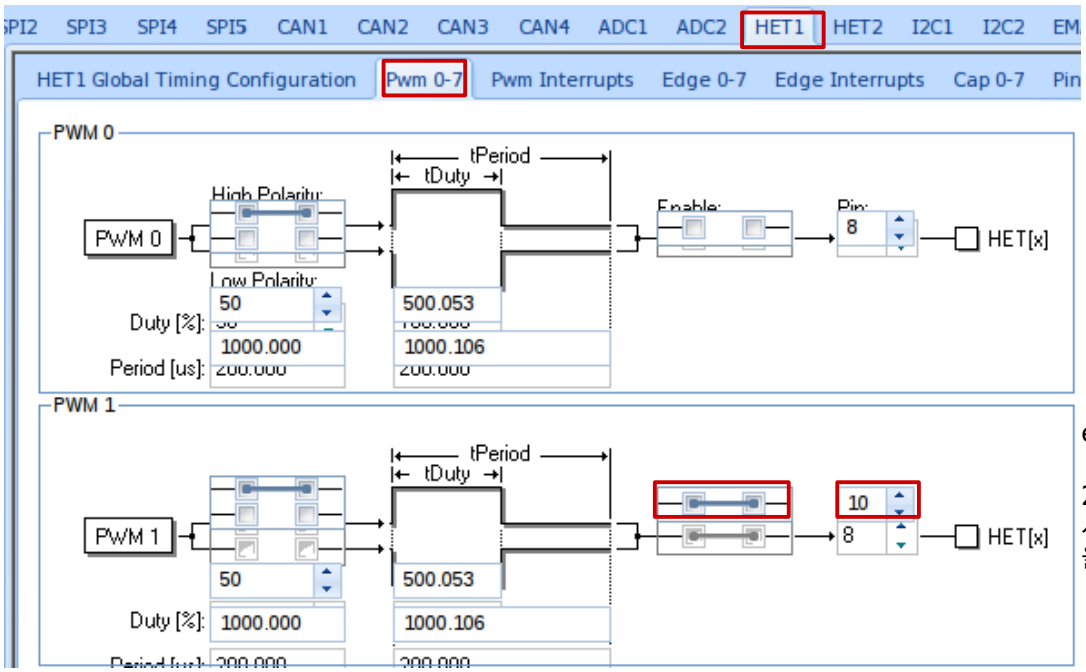
7. halcogen 설정하기. (TMS570LC4357ZWT → Driver Enble 탭)



1) Mark/Unmark all driver 해제

2) Enable HET1 driver 체크

8. (HET1 → Pwm 0-7 탭)



1) pwm1 을 enable

2) 10 번 pin 을 사용한 다는 것을 확인.

9. F5 를 누르면 Generate Code 를 한다. 그러면 CCS 의 경로를 설정한 프로젝트에 설정한 셋팅값이 프로그램 형태로 만들어 진다.

10. 이제 CCS 로 가서 프로그램을 시작하면 된다.

* 위의 pwm 핀은 HET[10]번 핀을 가리킨다.
  때문에 회로도를 보게 되면 D19 번 핀을 확인 할 수 있다.

CORETEX-R5 (hetPWM)

<code>

```
#include "HL_sys_common.h"
#include "HL_system.h"
#include "HL_het.h"


int main(void)
{
/* USER CODE BEGIN (3) */
    hetInit();

    while(1);

/* USER CODE END */

    return 0;
}
```

`1.hetInit()`로 드라이브.

```
      1). hetREG1->PFR = (uint32)((uint32) 6U << 8U)
                | ((uint32) 0U);
```

//PFR prescale pactor register 이다. 64 분주를 하게 하는 설정.

```
      2).  hetREG1->PCR = (uint32) 0x00000005U;
```

// PCR (Parity Control Register) parity 를 검사하지 않고 검사하지 않게 한다는 설정.

```
      3). ***(void)memcpy((void *)hetRAM1, (const void *) het1PROGRAM,
sizeof(het1PROGRAM));
```

(1) 제일 중요한 녀석이다.
(2) **기계어 단위→ 어셈블리 단위** 의 프로그램밍을 해야한다.
(3) ***instruction summary* 는 *12byte*** 인데 ***FPGA* 가 *16byte*** 로 처리되어 **<u>크기를 *16byte* 로 맞춤</u>**

(4) **het1PROGRAM → (해석이 중요)**

```
    /* CNT: Timebase
    *       - Instruction                  = 0
    *       - Next instruction             = 1
    *       - Conditional next instruction = na
    *       - Interrupt                    = na
    *       - Pin                          = na
    *       - Reg                          = T
    */
    {
        /* Program */   //13
        0x00002C80U,
        /* Control */
        0x01FFFFFFU,
```

```
    /* Data */
    0xFFFFFF80U,
    /* Reserved */
    0x00000000U
}
```

23.6.3.8 CNT (Count)

Syntax

```
CNT {
[brk={OFF | ON}]
[next={label | 9-bit unsigned integer}]
[reqnum={3-bit unsigned integer}]
[request={NOREQ | GENREQ | QUIET}]
[angle_count={OFF | ON}]
[reg={A | B | T | NONE}]
[comp ={EQ | GE}]
[irq={OFF | ON}]
[control={OFF | ON}]
max={25-bit unsigned integer}
[data={25-bit unsigned integer]
}
```

Figure 23-134. CNT Program Field (P31:P0)

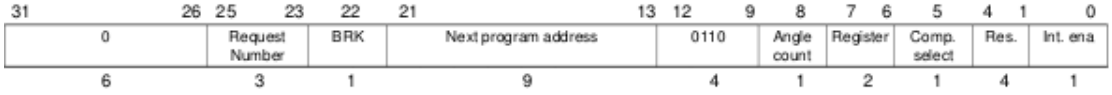
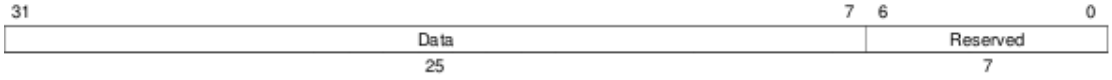


Figure 23-135. CNT Control Field (C31:C0)

Figure 23-136. CNT Data Field (D31:D0)



Cycles

One or two

One cycle (time mode), two cycles (angle mode)

Register modified

Selected register (A, B or T)

Description

This instruction defines a virtual timer. The counter value stored in the data field [D31:7] is incremented unconditionally on each execution of the instruction when in time mode (angle count bit [P8] = 0). When the count reaches the maximum count specified in the control field, the counter is reset. It takes one cycle in this mode.

In angle mode (angle count bit [P8] = 1), CNT needs data from the software angle generator (SWAG). When in angle count mode the angle increment value will be 0 or 1. It takes two cycles in this mode.

angle_count	Specifies when the counter is incremented. A value of ON causes the counter value to be incremented only if the new angle flag is set (NAF_global = 1). A value of OFF increments the counter each time the CNT instruction is executed. Default value for this field is OFF.
comp	When set to EQ the counter is reset, when it is equal to the maximum count. When set to GE the counter is reset, when it is greater or equal to the maximum count. Default: GE.
irq	ON generates an interrupt when the counter overflows to zero. The interrupt is not generated until the data field is reset to zero. If irq is set to OFF, no interrupt is generated. Default: OFF.
max	Specifies the 25-bit integer value that defines the maximum count value allowed in the data field. When the count in the data field is equal to max, the data field is reset to 0 and the Z system flag is set to 1.
data	Specifies the 25-bit integer value serving as a counter. Default: 0.

Table 23-73. Instruction Summary

Abbreviation	Instruction Name	Opcode	Sub-Opcode	Cycles[1]
ACMP	Angle Compare	Ch	-	1
ACNT	Angle Count	9h	-	2
ADCNST	Add Constant	5h	-	2
ADC	Add with Carry and Shift	4h	C[25:23] = 011, C5 = 1	1-3
ADD	Add and Shift	4h	C[25:23] = 001, C5 = 1	1-3
ADM32	Add Move 32	4h	C[25:23] = 000, C5 = 1	1-2
AND	Bitwise AND and Shift	4h	C[25:23] = 010, C5 = 1	1-3
APCNT	Angle Period Count	Eh	-	1-2
BR	Branch	Dh	-	1
CNT	Count	6h	-	1-2
DADM64	Data Add Move 64	2h	-	2
DJZ	Decrement and Jump if -zero	Ah	P[7:6] = 10	1
ECMP	Equality Compare	0h	C[6:5] = 00	1
ECNT	Event Count	Ah	P[7:6] = 01	1
MCMP	Magnitude Compare	0h	C[6] = 1	1
MOV32	Move 32	4h	C[5] = 0	1-2
MOV64	Move 64	1h	-	1
OR	Bitwise OR	4h	C[25:23] = 100, C5 = 1	1-3
PCNT	Period/Pulse Count	7h	-	1
PWCNT	Pulse Width Count	Ah	P[7:6] = 11	1
RADM64	Register Add Move 64	3h	-	1
RCNT	Ratio Count	Ah	P[7:6] = 00, P[0] = 1	3
SBB	Subtract with Borrow and Shift	4h	C[25:23] =110, C[5] = 1	1-3
SCMP	Sequence Compare	0h	C[6:5] = 01	1
SCNT	Step Count	Ah	P[7:6] = 00, P[0] = 0	3
SHFT	Shift	Fh	C[3] = 0	1
SUB	Subtract and Shift	4h	C[25:23] = 101, C[5] = 1	1-3
WCAP	Software Capture Word	Bh	-	1
WCAPE	Software Capture Word and Event Count	8h	-	1
XOR	Bitwise Exclusive-Or and Shift	4h	C[25:23] = 111, C[5] = 1	1-3

```
/* Program */     // 7, 10, 11, 14
//next program:1 opcode:0110 Angle counter:0 Register:10 compselect:0 Res:0000
ena:0
0x00002C80U

7bit:register
10bit:



/* Control */     // 0 ~ 24 까지.
//0000 0001 1111 1111 1111 1111 1111 1111
0x01FFFFFU

/* Data */        // 31 ~ 7 까지.
//1111 1111 1111 1111 1111 1111 1000 0000
0xFFFFFF80U
```