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

<펌웨어 프로그래밍>

2018.05.09 - 50일차

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

학생 – 안상재
sangjae2015@naver.com

<RTI(Real Time Interrupt) 초기화 및 각종 레지스터 셋팅 코드 분석>

1. main() 부분

```
int main(void)
{
    rtiInit();

    gioSetDirection(hetPORT1, 0xFFFFFFFF);
    rtiEnableNotification(rtiREG1, rtiNOTIFICATION_COMPARE0);

    _enable_IRQ_interrupt_();
    rtiStartCounter(rtiREG1, rtiCOUNTER_BLOCK0);

    while(1);

    return 0;
}
```

2. main() => rtiInit()

```
void rtiInit(void)
{
    /** - Setup NTU source, debug options and disable both counter blocks */
    // /* NTU1가 external timebase 로 사용됨 */
    rtiREG1->GCTRL = (uint32)((uint32)0x5U << 16U) | 0x00000000U;

    /** - Setup timebase for free running counter 0 */
    // RTIFRC0 will not be incremented on a failing external clock.
    // RTIUC0 clocks RTIFRC0.
    rtiREG1->TBCTRL = 0x00000000U;

    /** - Enable/Disable capture event sources for both counter blocks */
    // Capture of RTIUC1/ RTIFRC1 is triggered by capture event source 0.
    // Capture of RTIUC0/ RTIFRC0 is triggered by capture event source 0.
    rtiREG1->CAPCTRL = 0U | 0U;

    /** - Setup input source compare 0-3 */
    // Value will be compared with RTIFRC1
    // Value will be compared with RTIFRC1.
    // Value will be compared with RTIFRC0
    // Value will be compared with RTIFRC0
    rtiREG1->COMPCTRL = 0x00001000U | 0x00000100U | 0x00000000U | 0x00000000U;

    /** - Reset up counter 0 */
    rtiREG1->CNT[0U].UCx = 0x00000000U;

    /** - Reset free running counter 0 */
    rtiREG1->CNT[0U].FRCx = 0x00000000U;

    /** - Setup up counter 0 compare value
    *     - 0x00000000: Divide by 2^32
    *     - 0x00000001-0xFFFFFFFF: Divide by (CPUC0 + 1)
    */
    rtiREG1->CNT[0U].CPUCx = 7U;

    /** - Reset up counter 1 */
    rtiREG1->CNT[1U].UCx = 0x00000000U;

    /** - Reset free running counter 1 */
    rtiREG1->CNT[1U].FRCx  = 0x00000000U;

    /** - Setup up counter 1 compare value
```

```
    *     - 0x00000000: Divide by 2^32
    *     - 0x00000001-0xFFFFFFFF: Divide by (CPUC1 + 1)
    */
    rtiREG1->CNT[1U].CPUCx = 7U;

    /** - Setup compare 0 value. This value is compared with selected free running counter.
*/
    rtiREG1->CMP[0U].COMPx = 9375000U;

    /** - Setup update compare 0 value. This value is added to the compare 0 value on each
compare match. */
    rtiREG1->CMP[0U].UDCPx = 9375000U;

    /** - Setup compare 1 value. This value is compared with selected free running counter.
*/
    rtiREG1->CMP[1U].COMPx = 46875U;

    /** - Setup update compare 1 value. This value is added to the compare 1 value on each
compare match. */
    rtiREG1->CMP[1U].UDCPx = 46875U;

    /** - Setup compare 2 value. This value is compared with selected free running counter.
*/
    rtiREG1->CMP[2U].COMPx = 75000U;

    /** - Setup update compare 2 value. This value is added to the compare 2 value on each
compare match. */
    rtiREG1->CMP[2U].UDCPx = 75000U;

    /** - Setup compare 3 value. This value is compared with selected free running counter.
*/
    rtiREG1->CMP[3U].COMPx = 93750U;

    /** - Setup update compare 3 value. This value is added to the compare 3 value on each
compare match. */
    rtiREG1->CMP[3U].UDCPx = 93750U;

    /** - Clear all pending interrupts */
    rtiREG1->INTFLAG = 0x0007000FU;

    /** - Disable all interrupts */
    rtiREG1->CLEARINTENA = 0x00070F0FU;
}
```

3. main() => gioSetDirection()

- 해당 port 의 방향을 dir 인자를 통해 들어온 값으로 설정(1이면 출력, 0이면 입력)

```
void gioSetDirection(gioPORT_t *port, uint32 dir)
{
    port->DIR = dir;
}
```

4. main() => rtiEnableNotification()

```
void rtiEnableNotification(rtiBASE_t *rtiREG, uint32 notification)
{
    rtiREG->INTFLAG = notification; // compare0 에 관련된 어떠한 인터럽트도 걸려있지 않음
    rtiREG->SETINTENA   = notification; // compare interrupt 0 이 enabled 됨
}
```

5. main() => _enable_IRQ_interrupt()

```
_enable_IRQ_interrupt_

        cpsie i
        bx    lr

        .endasmfunc
```

6. main() => rtiStartCounter()

```
void rtiStartCounter(rtiBASE_t *rtiREG, uint32 counter)
{
    rtiREG->GCTRL |= ((uint32)1U << (counter & 3U)); // Counter block 0 is running
}
```

7. 인터럽트 핸들러 : main() => rtiNotification()

- 1초에 한번씩 핸들러로 들어오게 되고, 0x1을 xor 함으로써 계속 HED1_00 을 반전시킴.

```
void rtiNotification(rtiBASE_t *rtiREG, uint32 notification)
{
    gioSetPort(hetPORT1, gioGetPort(hetPORT1) ^ 0x00000001);
}
```

- gioGetPort() : 현재 port 의 값 상태를 반환함.

```
Uint32 gioGetPort(gioPORT_t *port)
{
    return port->DIN;
}
```

- gioSetPort() : 현재 port 의 출력에 값을 내보냄.

```
void gioSetPort(gioPORT_t *port, uint32 value)
{
    port->DOUT = value;
}
```