7 Timer Interrupt Programming

내용

- Interrupt controller
  - Interrupt controller 소개
  - Interrupt controller 주요 레지스터
- Timer
  - Timer 소개
  - Timer 주요 레지스터
- 실습 프로그램: start.S, timer.c, handlers.c, handlers.h

Interrupt Controller 소개

- Interrupt controller(INTC)는 여러 device로부터 전달되는 interrupt들을 processor의 IRQ 혹은 FIQ에 전달하는 역할을 함
- 여러 device로부터 전달되는 interrupt들은 MIRn(n=0 to 3) 레지스터를 사용하여 각각 독립적으로 enable 혹은 disable할 수 있음
- 여러 device로부터 전달되는 interrupt들은 ILRm(m=0 to 127) 레지스터를 사용하여 processor의 IRQ 혹은 FIQ로 전달할 수 있음
- AM335x의 INTC는 총 128(0-127)개의 device로부터 전달되는 interrupt를 처리할 수 있음

Interrupt Controller 주요 레지스터

레지스터 이름	주소	레지스터 설명
INTC_MIRn	0x48200000 + 0x0084 + (0x20 * n)	Interrupt mask register n (n=0 to 3)
INTC_CONTROL	0x48200048	New interrupt agreement register

INTC_MIRn: Interrupt Mask Register n

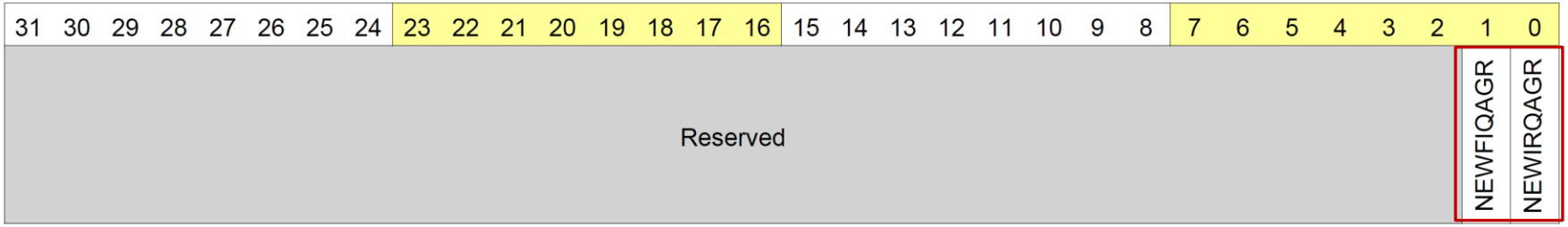
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIR																															

Bits	Field Name	Description	Type	Reset
31:0	MIR	Interrupt mask 0x1: The interrupt is masked 0x0: The interrupt is unmasked	RW	0xFFFFFFFF

- 4개의 MIR0, MIR1, MIR2 및 MIR3 register가 128개의 interrupt에 해당함 (MIR0: 0-31, MIR1: 32-63, MIR2: 64-95, MIR3: 96-127)
  - 0x0: 해당 interrupt이 enable됨
  - 0x1: 해당 interrupt이 disable됨

참고: 이 register는 interrupt 초기화 시 원하는 interrupt를 enable 혹은 disable할 때 사용함 (DMTimer1=67)

INTC_CONTROL: New Interrupt Agreement Register



- NEWIRQARG: 0x1을 write하면 IRQ output을 reset하고 새 IRQ 생성을 enable함
- NEWFIQAGR: 0x1을 write하면 FIQ output을 reset하고 새 FIQ 생성을 enable함

참고: 이 register는 발생한 interrupt를 처리한 후 해당 IRQ 혹은 FIQ를 reset하고 새 interrupt를 받기 위하여 사용함

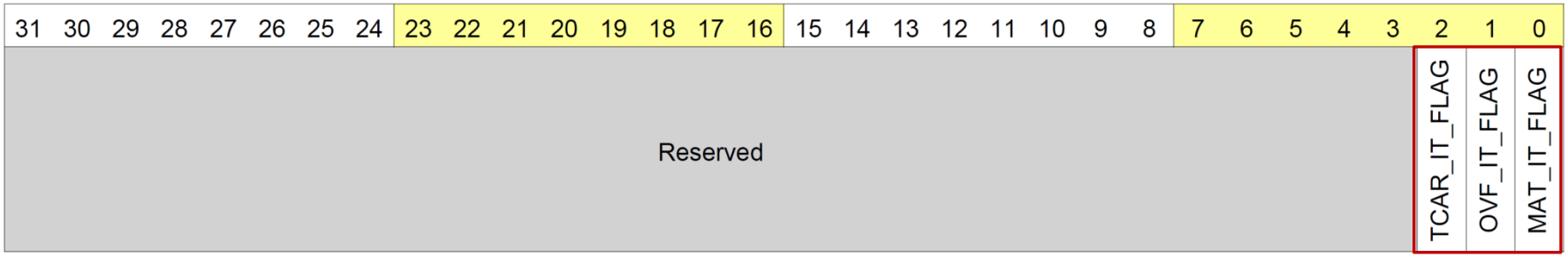
Timer 소개

- AM335x는 8개의 Timer인 DMTimer0-7을 제공함
- 각 DMTimer는 32-bit upward counting 기능 제공
- 각 DMTimer는 one-shot/autoreload mode 및 stop/start 기능이 있음
- 각 DMTimer는 MPU subsystem에 capture/ overflow/match interrupt를 발생시킬 수 있음
- 각 DMTimer는 system clock 혹은 32.768 KHz clock을 사용함 (DMTimer0는 32.768 KHz clock만 사용)
- DMTimer1은 정확한 1 ms(=10^{-3} sec) tick interrupt를 제공하기 위한 기능을 가지고 있음 (우리 실습 시 사용)

Timer 주요 레지스터 (DMTimer1)

레지스터 이름	주소	레지스터 설명
TISR	0x44e31018	Timer status register
TIER	0x44e3101c	Timer interrupt enable register
TCLR	0x44e31024	Timer control register
TCRR	0x44e31028	Timer counter register
TLDR	0x44e3102c	Timer load register
TPIR	0x44e31048	Timer positive increment register
TNIR	0x44e3104c	Timer negative increment register

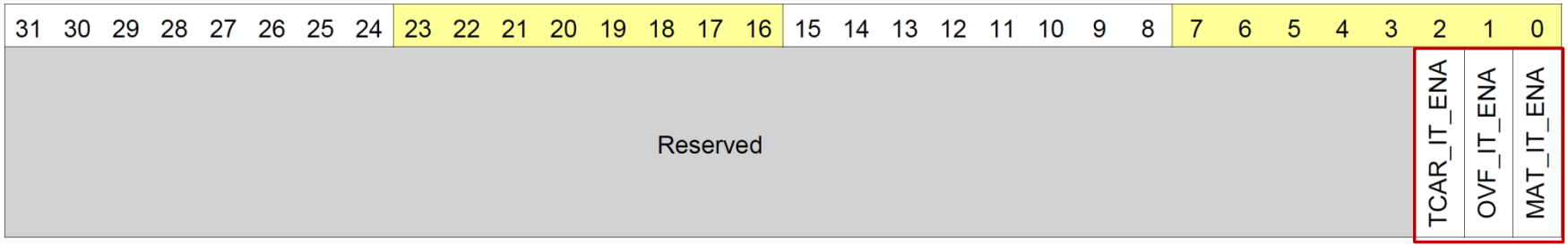
TISR: Timer Status Register



- TCAR_IT_FLAG(=Pending capture interrupt status), OVF_IT_FLAG(=Pending overflow interrupt status), MAT_IT_FLAG(=Pending match interrupt status) 이 있음
  - read 0x0: no interrupt pending
  - read 0x1: interrupt pending
  - write 0x0: status unchanged
  - write 0x1: status bit cleared

참고: 이 register는 timer interrupt 초기화 시(write 0x1), interrupt handler에서 해당 interrupt 확인 시(read 0x1), interrupt handler 수행 후 해당 interrupt를 clear할 때(write 0x01) 사용함

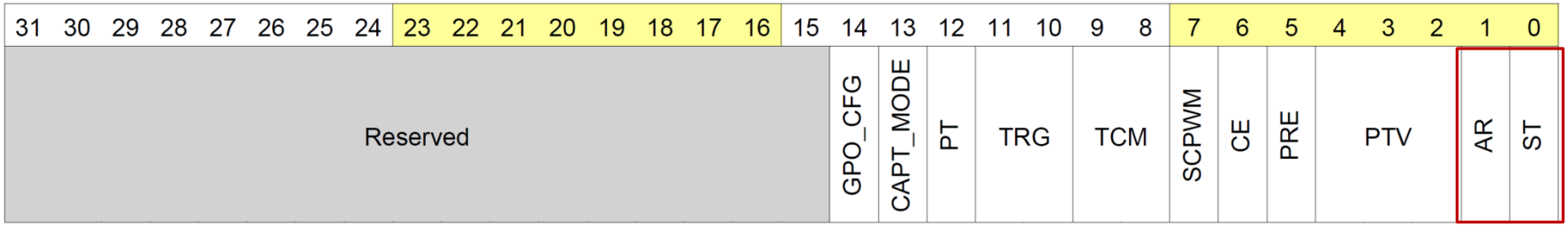
TIER: Timer Interrupt Enable Register



- TCAR_IT_ENA(=Enable capture interrupt), OVF_IT_ENA(=Enable overflow interrupt), MAT_IT_ENA(=Enable match interrupt)
  - write 0x0: disable interrupt
  - write 0x1: enable interrupt

참고: 이 register는 timer interrupt 초기화 시 capture/overflow/match interrupt를 enable(write 0x1)할 때 사용함

TCLR: Timer Control Register



- ST bit
  - write 0x0: stop timer control
  - write 0x1: start timer control
- AR bit
  - write 0x0: one-shot mode control
  - write 0x1: autoreload mode control

참고: 이 register는 timer를 stop시키거나(ST bit write 0x0) start 시키고(ST bit write 0x1) autoreload mode설정 시 (AR bit write 0x1) 사용함

TCRR/TPIR/TNIR/TLDR

- TCRR: Timer Counter Register:
  - upward counting하며 현재 timer counter의 값을 저장 (1ms timer: write 0xfffffe0)
- TPIR: Timer Positive Increment Register
  - 1ms timer를 구현하기 위한 값 (1ms timer: write 232000)
- TNIR: Timer Negative Increment Register
  - 1ms timer를 구현하기 위한 값 (1ms timer: write -768000)
- TLDR: Timer Load Register
  - 초기 timer에 load되는 값을 저장 (1ms timer: write 0xfffffe0)

실습 프로그램

- start.S: timer interrupt 처리를 위한 적절한 vector 및 C 언어로 작성된 handler 함수를 부르는 assembler 프로그램 (6 Exception 프로그래밍 참고)
- timer.c: timer interrupt를 시험하는 함수 main
- handlers.c: timer interrupt handler, timer interrupt 발생을 위한 초기화 함수, timer interrupt start 및 stop 함수
- handlers.h: timer interrupt 처리에 필요한 register에 대한 주소를 정의하는 macro

timer.c

```
int main(int argc, char *argv[])
{
  char input[80];

  for (;;) {
    UART_printf("========================================\n");
    UART_printf("Timer Interrupt (%s mode)\n", get_current_mode());
    UART_printf("========================================\n");
    UART_printf("1. Initialize timer1\n");
    UART_printf("2. Enable timer1\n");
    UART_printf("3. Disable timer1\n");
    UART_printf("4. Print clock\n");
    UART_printf("5. Quit\n");
    UART_printf("Select one: ");

    ...
}
```

handlers.c

```
// ================================
// exception handler for interrupt

volatile unsigned int clock = 0;

void handle_irq(unsigned int *sp,
   unsigned int spsr)
{
  // If DMTIMER1 TISR has overflow
   event, increment clock.
  ...

  // INTC_CONTROL: reset IRQ
   generation and enable new IRQ.
  ...
}
```

```
void TIMER_init(void)
{
  ...
}

void TIMER_enable(void)
{
  ...
}

void TIMER_disable(void)
{
  ...
}
```

handlers.h

```
#define IO_READ(addr)              (*(volatile
    unsigned int *)(addr))
#define IO_WRITE(addr, val)        (*(volatile
    unsigned int *)(addr) = (val))
#define IO_RCW(addr, mask, val) IO_WRITE(addr,
    (IO_READ(addr) & ~(mask)) | ((val) &
    (mask)))

// ========================================
// Dual Module Timer 1 Registers.

#define DMTIMER1_BASE           0x44e31000
#define TISR                    0x18
#define TIER                    0x1c
#define TCLR                    0x24
#define TCRR                    0x28
#define TLDR                    0x2c
#define TPIR                    0x48
#define TNIR                    0x4c
#define TSICR                   0x54
```

```
// ========================================
// Interrupt controller registers.

#define INTCPS_BASE             0x48200000
#define INTC_CONTROL            0x48
#define INTC_MIR0               0x84
#define INTC_MIR1               0xa4
#define INTC_MIR2               0xc4
#define INTC_MIR3               0xe4

// ========================================
```

참고 문헌

1. ARM Limited, ARM® Architecture Reference Manual, ARM DDI 0100I, July 2005. (Chapter A2.6)
2. ARM Limited, ARM® Architecture Reference Manual ARM®v7-A and ARM®v7-R Edition, ARM DDI 0406B, July 2009. (Chapter B1.6)
3. Texas Instruments, AM335x ARM® Cortex™-A8 Microprocessors (MPUs) Technical Reference Manual, SPRUH73J, December 2013. (Chapter 6, 8, 20)