AN

[MiDAS Family] Training Guide for Lab. with GENTOS

MiDAS Family with HERA T2K Application Board

V1.2

March 2006

Contents

- Lab.1 : Getting Started
  1) Download the Program “GENTOS”
  2) Program Run & MDS Setting
  3) How to Use the Program “GENTOS”
- Lab.2 : How to Use the GENTOS Program
  1) Description
  2) Breakpoint Marking Setting
  3) Add the Variable to the “Watch Window”
- Lab.3 : How to Use Timer0 Interrupt
  1) Example List
  2) Description
  3) Code
- Lab.4 : How to Use ADC Interrupt
  1) Example List
  2) Description
  3) Code
- Lab.5 : How to Use UART Interrupt
  1) Example List
  2) Description
  3) Code

- Lab.1 & Lab.2 are only for showing the how to use GENTOS program & MDS equipment. User don’t care about source code.
- Lab.3, Lab.4, and Lab.5 are for showing the how to use the peripheral of MCU. In more detail information, please refer to the brief/full manual from download center of CORERIVER Homepage (www.coreriver.com)

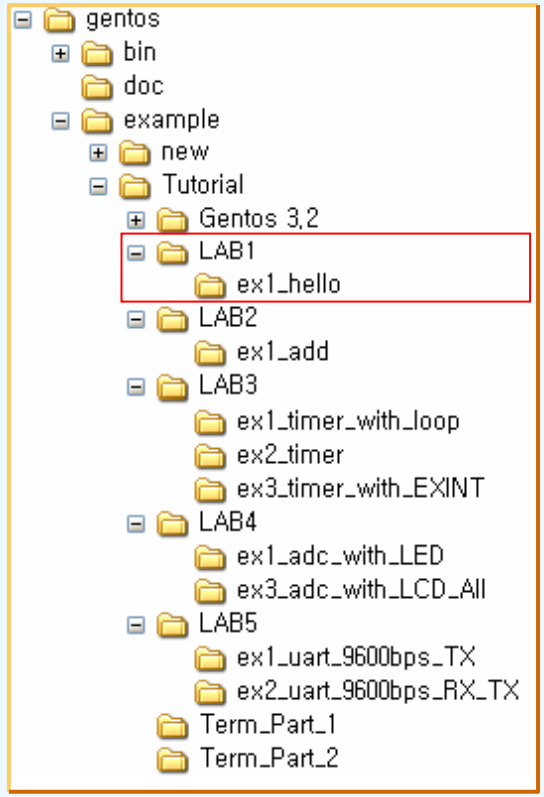
Lab.1 : Getting Started

- Lab.1 : 1) Download the Program “GENTOS” (Slide 1 of 5)
- Lab.1 : 2) Program Run & MDS Setting (Slide 2 of 5)
- Lab.1 : 3) How to Use the Program “GENTOS” (Slide 3 of 5)

Lab.1 : 1) Example List (1 of 5)

- Objective : Getting Started
  - ✓ How to download the program “GENTOS”.
  - ✓ How to use the MDS (Micro-controller Development System) equipment “GENSYS52” & program “GENTOS” for MiDAS Family.
- Example List
  - ✓ hello
- Example : hello
  - ✓ This example is for getting started and for learning MDS & program usage.
  - ✓ User can not be care about the contents of the example code.
  - ✓ We manipulate about the example code for MCU (MiDAS Family) from next Lab.2 to Lab.5.
  - ✓ Execution : 16 Characters X 4 lines “Character LCD” Output at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - LCD Output : HELLO

- C:\gentos\example\Tutorial
- Tutorial Examples is for MiDAS Family.

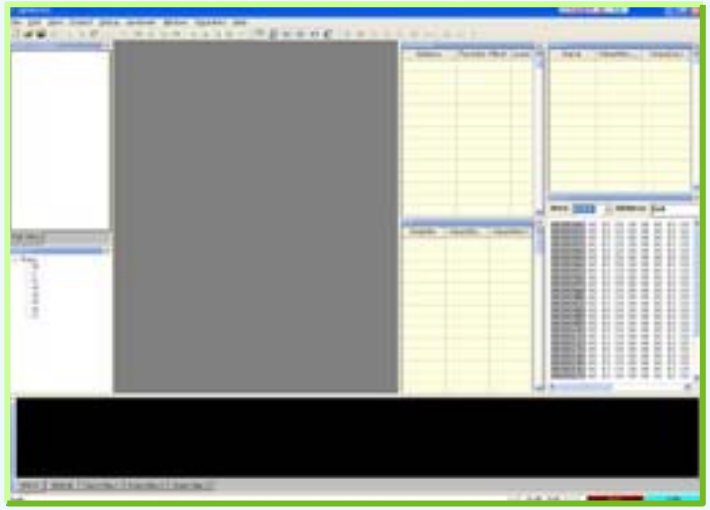


Lab.1 : 2) Program & Lab. Setting (2 of 5)

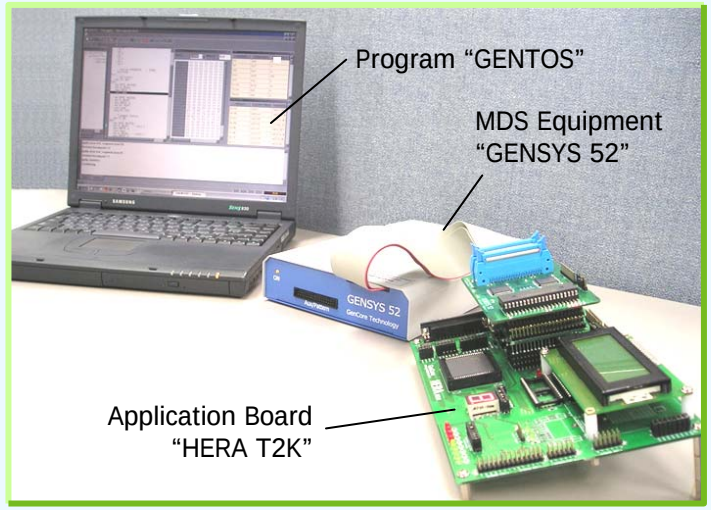
- Please, install the program “GENTOS”.
- We recommend the install directory : “C:\gentos”.
- Program Run.
  - ✓ Windows “Start” Menu → “Program (P)” Tab Menu → “GENTOS” Group Menu → “Gentos” Program Icon Click.

- Manual to be able to download
  - ✓ Brief Manual of GENTOS
  - ✓ Brief Manual of MiDAS Family
  - ✓ Full Manual of MiDAS Family
  - ✓ Brief Manual of HERA T2K
- Lab. Setting
  - ✓ PC Program : GENTOS
  - ✓ MDS Equipment : GENSYS52
  - ✓ Application Board : HERA T2K

[GENTOS Program]



[MDS Setting for Lab.]



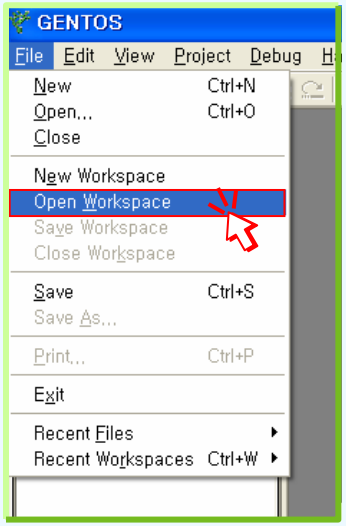
Lab.1 : 3) How to Use the GENTOS (3 of 5)

- Open the workspace file (= project file).
  - ✓ After the "GENTOS" program is installed, examples are supported for MiDAS Family and tutorial.
  - ✓ Click the top menu "File → Open Workspace".
  - ✓ Location of Workspace File : "`C:\gentos\example\Tutorial\LAB1\hello\hello.gts`"

- Power "ON" for MDS & HERA T2K
  - ✓ First, MDS Equipment Power ON
  - ✓ And HERA T2K Application Board Power ON

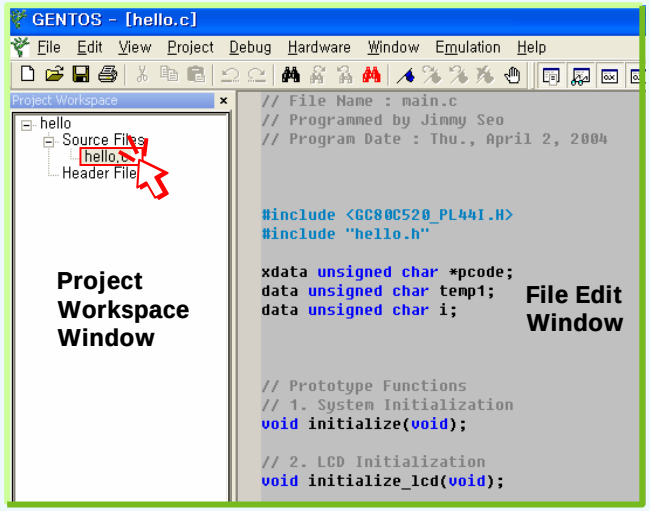
- Power "OFF" for MDS & HERA T2K
  - ✓ First, HERA T2K Application Board Power OFF
  - ✓ And MDS Equipment Power OFF

(1) Menu : Open Workspace



Open "`hello.gts`"

(2) Open C Source File : `hello.c`

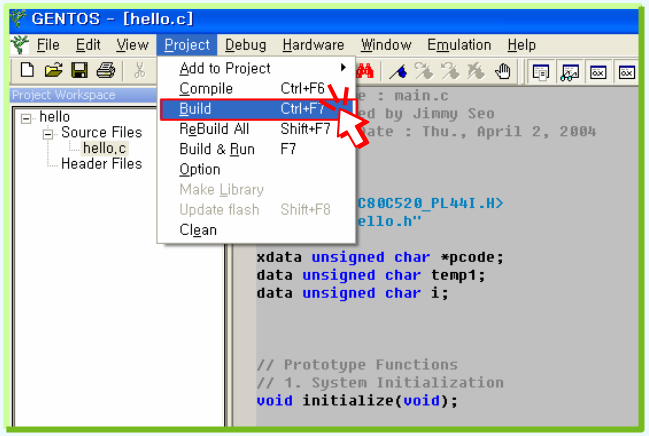


Lab.1 : 3) How to Use the GENTOS (4 of 5)

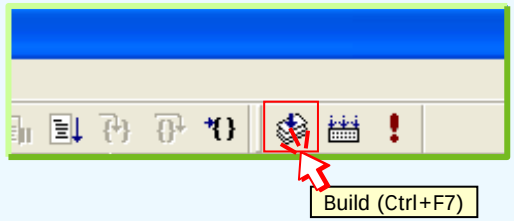
- Top Menu : "Project"
  - ✓ Compile : Control + F6 Key
  - ✓ Build : Control + F7 Key
  - ✓ Build & Run : F7 Key
- Tool Button
  - ✓ : Compile (Ctrl + F6)
  - ✓ : Build (Ctrl + F7)
  - ✓ : Build & Run (F7)
  - ✓ : Go (Run) (F5)

- Please, build the file.
  - ✓ Click the top menu "Project → Build".
  - ✓ or click the tool button " Build".

(1) Menu : Build



(2) Tool Button : Build



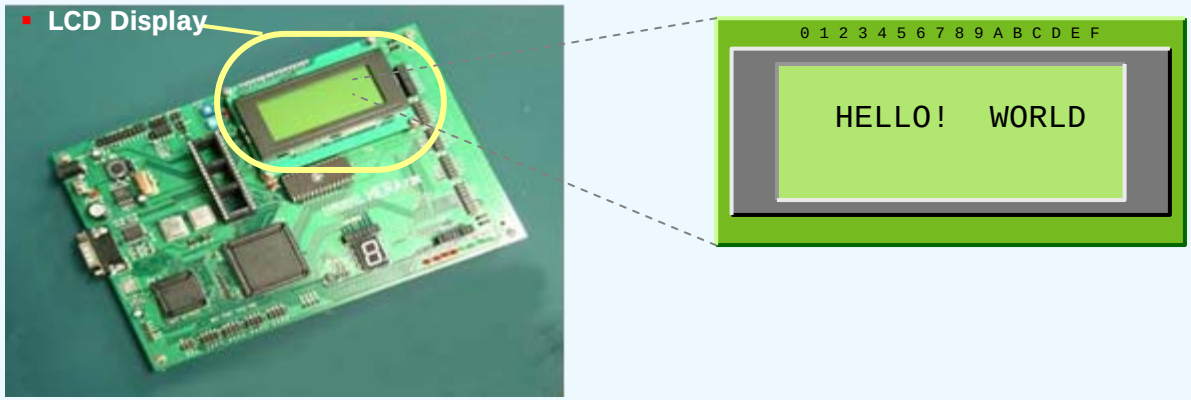
[Description : Top Menu "Project"]

1. Compile (Ctrl+F6) : Compile the File
2. Build (Ctrl+F7) : Compile & generating the HEX File.
3. Build & Run (F7) : Build & Run with MDS Equipment

Lab.1 : 3) How to Use the GENTOS (5 of 5)

- Run the workspace file.
  - ✓ Click the top menu “Debug → Go” (F5 key).
  - ✓ Or click the tool button “Go”. 
- Check the result at demo at HERA T2K application Board as below.
  - ✓ Display the string “HELLO” at LCD.

◆ **HERA T2K Application Board**



◆ Refer to the Brief Manual of HERA T2K Application Board (BM-HERAT2K-VXX.pdf) for more detail information.

◆ **Note** : This demo program is only the delay function without timer interrupt function for demo.
We recommend that user must use the timer interrupt function if the timing control is needed.

Memo

Lab.2 : How to Use the GENTOS Program

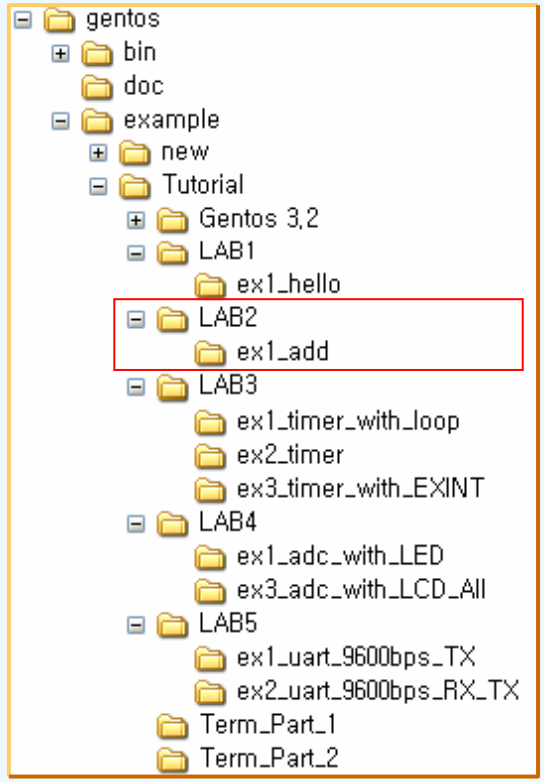
- Lab.2 : 1) Example List (Slide 1 of 8)

- Lab.2 : 2) Tutorial Guide of Lab. (Slide 2 of 8)
  - Description (Slide 2 of 8)
  - Breakpoint Marking Setting (Slide 5 of 8)
  - Add the Variable to the “Watch Window” (Slide 7 of 8)

Lab.2 : 1) Example List (1 of 8)

- Objective : How to Use the GENTOS program
- Example List
  - ✓ add
- Example : add
  - ✓ This example shows how to use the GENTOS program (Build, Run, breakpoint, step, stop, and stop debugging).
  - ✓ Execution : Display the result for addition of program to LCD (Port A) Output at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - Operation Result Display : LCD Display (Delay Function)

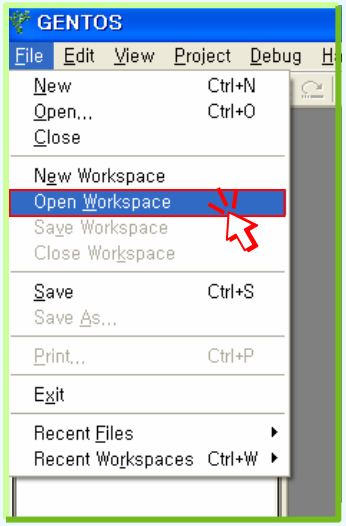
- `C:\gentos\example\Tutorial`
- Tutorial Examples is for MiDAS Family.



Lab.2 : 2) Tutorial Guide of Lab. (2 of 8)

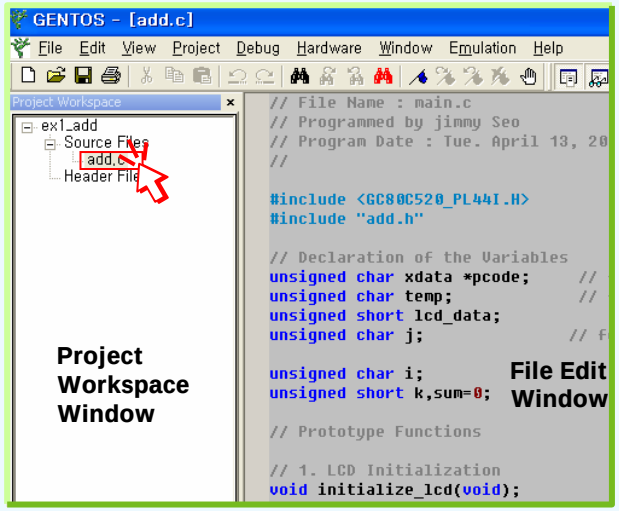
- Open the workspace file (= Project File)
  - ✓ Click the top menu "File → Open Workspace".
  - ✓ Location of Workspace File : "`C:\gentos\example\Tutorial\LAB2\ex1_add\add.gts`"
  - ✓ Click the top menu "File → Open Workspace".
- Open the source file "add.c" by double click at the project window (Left).

(1) Menu : Open Workspace



Open "add.gts"

(2) Open C Source File : add.c



Lab.2 : 2) Tutorial Guide of Lab. (3 of 8)

- Please, insert the code for addition.
  - ✓ 1 + 2 +3 ... + 10

(1) Original Source File (add.c)

```
// Sub Function : 10.
void run (unsigned char addr) {

        unsigned short k, sum = 0;

        // Please, insert the code.
        // Add the number : 1 + 2 + ... + 10

        address_lcd(addr);                      } LCD Output

        Print_INT(&k);
        delay(600);


        address_lcd(addr + 0x10);               } LCD Output

        Print_INT(&sum);
        delay(600);

}
```

(2) Modified Source File (add.c)

```
// Sub Function : 10.
void run (unsigned char addr) {

        unsigned short k, sum=0;

        // Added Code
        for (k=0; k<=10; k++) {
                sum += k;
```

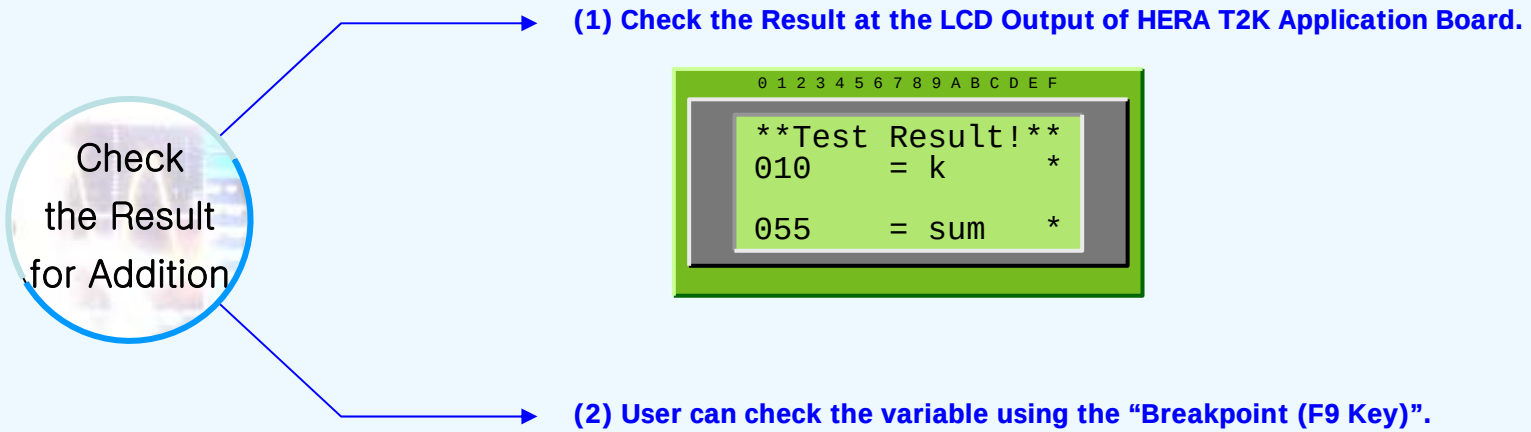
Code Added

```
                address_lcd(addr);

                Print_INT(&k);
                delay(600);


                address_lcd(addr + 0x10);

                Print_INT(&sum);
                delay(600);

        }
```

Code Added

```
}
```

- Result Output for Addition
  - ✓ User can check the result for addition at LCD of HERA T2K Application Board.
  - ✓ If the bug(s) at source file exists, user must correct the source file.
  - ✓ At this time, user can easily debug and execute the emulation using MDS equipment.
  - ✓ User can use the "breakpoint" (F9 Key) of GENTOS program and check the current status.
    - "Breakpoint" Making Setting
    - Variable addition at "Watch Window".



- Breakpoint Marking Setting
  - ✓ First, please place the mouse cursor at the code line wanted breakpoint.

✓ And user can set the breakpoint.

- Top Menu : "Debug → Breakpoint" (F9 Key)
- Pop-up Menu : Click the mouse's right button and select the "Toggle Breakpoint".

(1) Breakpoint Marking Setting

Breakpoint Marking

Breakpoint marking

```
// Sub Function : 10.
void run (unsigned char addr) {

        unsigned short k, sum=0;

        // Added Code
        for (k=0; k<=10; k++) {
                sum += k;
        //

                address_lcd(addr);

                Print_INT(&k);          <- Variable "k" Output at LCD.
                delay(600);


                address_lcd(addr + 0x10);

                Print_INT(&sum);        <- Variable "sum" Output at LCD.
                delay(600);

        }

}
```

Lab.2 : 2) Tutorial Guide of Lab. (6 of 8)

- Power "ON" for MDS & HERA T2K
  - ✓ First, MDS Equipment Power ON
  - ✓ And HERA T2K Application Board Power ON
- Build, and Run. Or Build & Run.
  - ✓ Top Menu #1 : "Project → Build" (Ctrl + F7 Key), and "Debug → Go" (F5 Key)
  - ✓ Top Menu #2 : "Project → Build & Run" (F7 Key),

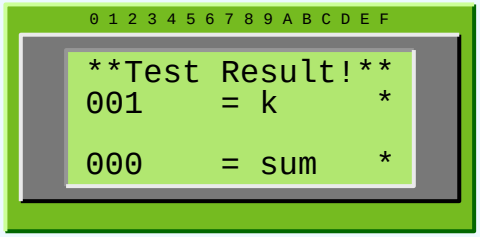
✓ If the program is executed, the program is stopped at the breakpoint as below.

Stop point by breakpoint

```
        for (k=0; k<=10; k++) {
              sum += k;
        //

              address_lcd(addr);

              Print_INT(&k);
              delay(600);


              address_lcd(addr + 0x10);

              Print_INT(&sum);
              delay(600);

        }
}
```

* When user execute the "Run (F5 Key)", the program is stopped at the next breakpoint marking.

- Initial LCD Output at HERA T2K Application Board.



[LCD Output]

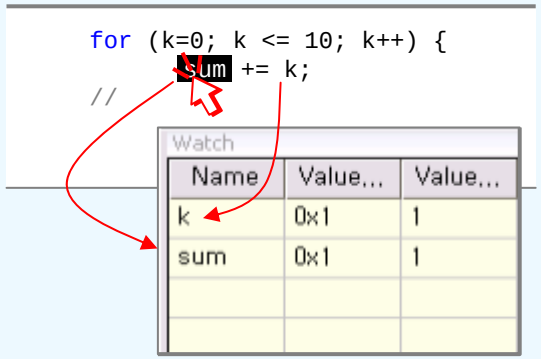
1. The program uses the function "display()" for Sting output.
2. The program uses the function "`Print_INT()`" for the variable k & sum output.

- If user executes the command "Go" (F5 key) 3 times.

```
        for (k=0; k<=10; k++) {
                sum += k;
        //

                address_lcd(addr);

                Print_INT(&k);
                delay(600);


                address_lcd(addr + 0x10);

                Print_INT(&sum);
                delay(600);

        }

}
```

[Variable Setting at "Watch Window"]

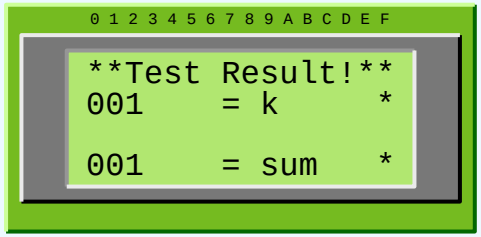
1. Select the variable as below. And the click the mouse' right button.
   Select the "Add to Watch" from pop-up menu.



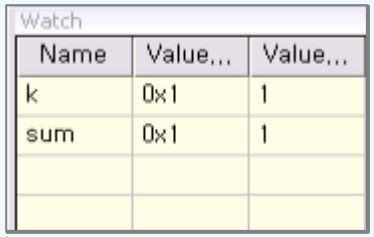
Lab.2 : 2) Tutorial Guide of Lab. (8 of 8)

- At this time, the variables' value at the "Watch Window" are as below.

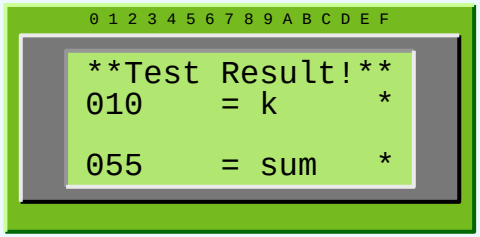
(1) LCD Output



(2) Watch Window of GENTOS Program



- If user repeatedly executes the command "Go" (F5 key), the result is as below.



Memo

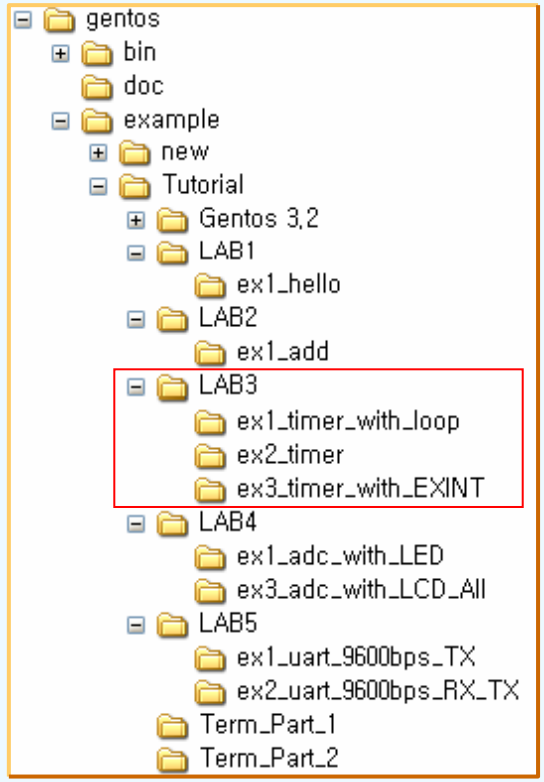
Lab.3 : How to Use Timer0 Interrupt.

- Lab.3 : 1) Example List (Slide 1 of 7)
- Lab.3 : 2) Description
  - SFR : TMOD (Slide 3 of 7)
  - Timer0 Mode Setting (Slide 3 of 7)
  - Time Calculation (Slide 3 of 7)
  - Timer Interrupt Setting / Timer Run (Slide 4 of 7)
- Lab.3 : 3) Code
  - Main Function (Overview) (Slide 5 of 7)
  - Sub Function : `initialize()` (Slide 6 of 7)
  - Interrupt Function : `timer0_int()` (Slide 7 of 7)

Lab.3 : 1) Example List (1 of 7)

- Objective : How to use Timer0 Interrupt
- Example List
  - ✓ ex1_timer_with_loop
  - ✓ ex2_timer
- Example : ex1_timer_with_loop
  - ✓ This example shows how to use timer0 interrupt with delay function.
  - ✓ Execution : 7-segment Display (Port B) and LED (Port C) at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - Timer0 Interrupt Interval : 0.25 msec (250 usec)
    - Timer0 Mode : Mode 2 (8-bit, Auto-reload)
    - Display Mode Changing : 1 sec
    - Display Value Update : delay function (`for` Loop)

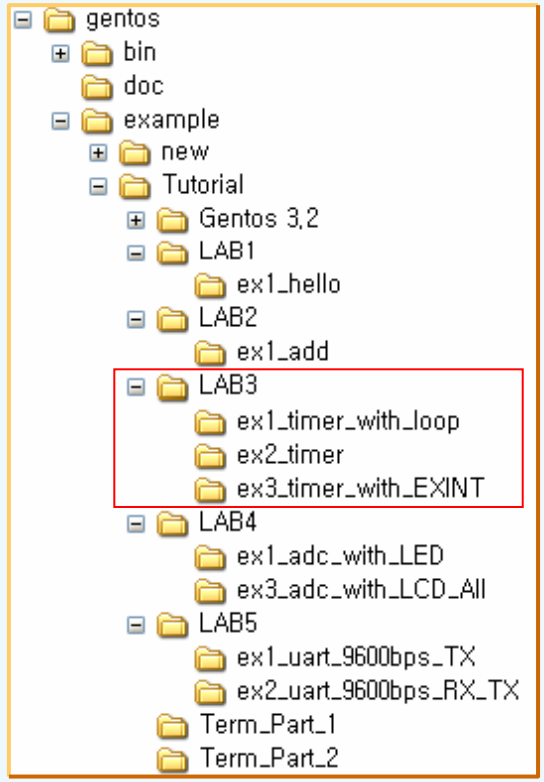
- `C:\gentos\example\Tutorial`
- Tutorial Examples is for MiDAS Family.



Lab.3 : 1) Example List (2 of 7)

- Example : ex2_timer
  - ✓ This example shows how to use timer0 interrupt.
  - ✓ Execution : 7-segment Display (Port B)
    at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - Timer0 Interrupt Interval : 0.25 msec (250 usec)
    - Timer0 Mode : Mode 2 (8-bit, Auto-reload)
    - 7-segment Display Update : 0.5 sec
- Example : ex3_timer
  - ✓ This example shows how to use timer0 interrupt with remote controller.
  - ✓ Execution : 7-segment Display (Port B)
    at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - Timer0 Mode : Mode 1 (16-bit)
    - 7-segment Display Update : by remote controller

- C:\gentos\example\Tutorial
- Tutorial Examples is for MiDAS Family.



Lab.3 : 2) Example 2 : Description (3 of 7)

- The objective of Lab.3 is how to use timer0 interrupt and function.
- We explain the example : ex2_timer.
- This example only uses the timer0 interrupt without delay function.
- Condition
  - ✓ System Clock (F_{OSC}) : 12 MHz
  - ✓ Timer0 Interrupt Interval : 0.25 msec
  - ✓ Timer0 Mode : Mode 2
  - ✓ 7-segment Display Update : 0.5 sec
- Time Calculation of Timer/Counter0.
  - ✓ Mode2 : 8-bit Auto-reload

When C/T = 0 (Default),

$$\text{Time[sec]} = \left(\frac{F_{OSC}}{12}\right)^{-1} \times \left(2^8 - \text{TH0}\right)$$

  - ✓ When System Clock (F_{OSC}) is 12 MHz and the TH0 for reload is 0x06, the timer0 interrupt interval is 0.25 msec.

- Mode Setting for Timer0
  - ✓ SFR Register : TMOD (Refer below SFR)
  - ✓ Mode 2, 8-bit T/C with 8-bit Auto-reload.

```
TMOD = 0x02;     // Timer0, Mode2
```

- SFR (Special Function Register) : TMOD

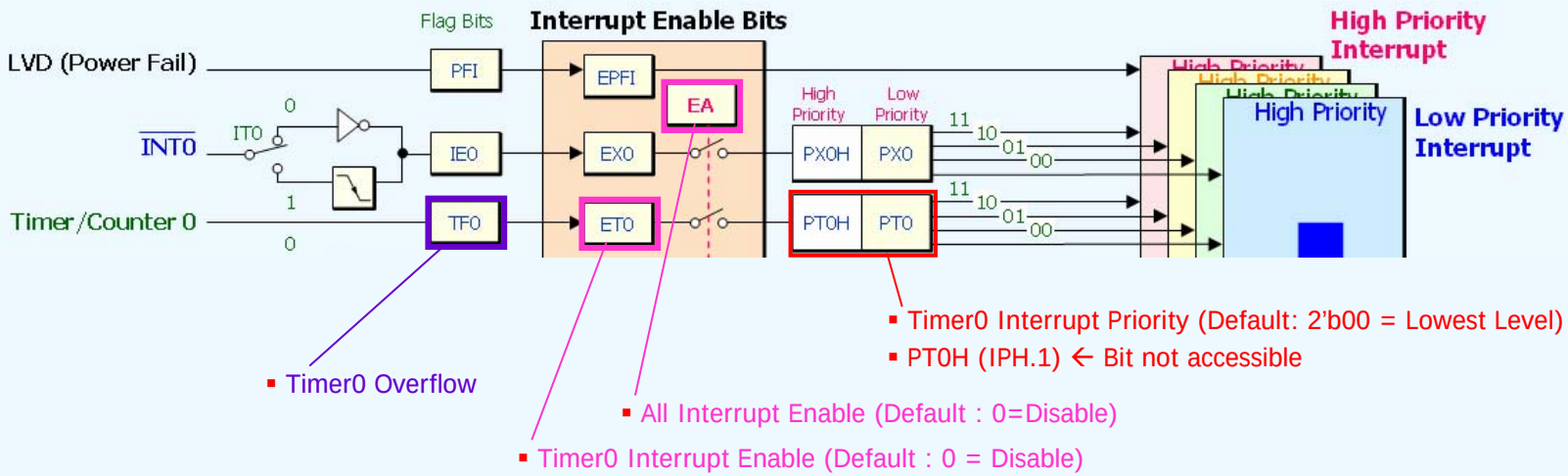
■ **TMOD** (89h) : Timer/Counter 0 Mode Control Register

GATE	C/T	M1	M0	GATE	C/T	M1	M0
R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)
Timer1 Mode Control				Timer0 Mode Control			

- GATE : Timer gate control.
- C/T : Timer Counter/Timer select. When set, counter by TX pin.
- M1, M0 : Timer mode selection.
  - [0,0] : Mode0, 13-bit T/C (Timer/Counter)
  - [0,1] : Mode1, 16-bit T/C
  - [1,0] : Mode2, 8-bit T/C with auto-reload
  - [1,1] : Mode3, Two 8-bit T/C

Lab.3 : 2) Example 2 : Description (4 of 7)

- Timer0 Interrupt Setting
  - ✓ User must set the wanted interrupt enable flag to "1".
  - ✓ Because we want to use Timer0 interrupt, set the "Timer0 interrupt enable flag (ET0)" to "1".
  - ✓ Remember that user must set the "All Interrupt Enable (EA)" to "1" to use the set interrupt(s).
  - ✓ User can set the interrupt priority.
  - ✓ Refer to below Figure and description.
- In the Example,
  - ✓ For the relative code for TF0, refer to timer0 interrupt function : `timer0_int()`
  - ✓ For the relative code for ET0 and EA, refer to initial setting function : `initialize()`
  - ✓ User can insert the relative code for priority flags (PT0 and PT0H) to initial setting function.
  - ✓ For Timer0 run, user must insert the "TR0 = 1".



- **Reference** : Brief Manual of MiDAS Family (BM-MiDAS1.0-VXX.pdf) : 6-12. Interrupt Functional Description
- **Note** : The yellow color of the upper flags is "bit-accessible", and white color is "bit not accessible".

Lab.3 : 3) Example 2 : Code (5 of 7)

◆ **Main Function**

```
// Header File : MiDAS1.0 Family
#include <GC80C520_PL44I.H>


void main(void) {

     initialize();

     TR0 = 1;       // Timer0 Run

     while(1) {
          temp = 0;
     }

}
```

◆ **Sub Function : Initialization**

```
void initialize(void)
```

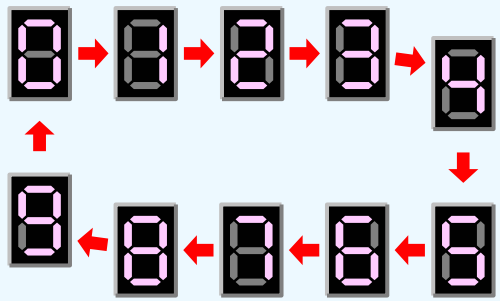
- Timer0 Setting
- Initialization of Global Variable(s)

Timer0 Interrupt Interval : 0.25 msec

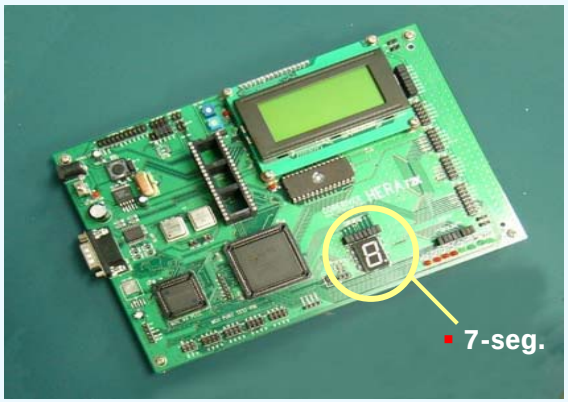
◆ **Interrupt Function**

```
void timer0_int(void) interrupt TF0_VECTOR
```

- Timer0 Interrupt Interval : 0.25 msec (250 usec)
- Global variable "loop_cnt" is incremented "+1" at each interrupt.
- `if (loop_cnt == 2000)` ➔ 0.25 msec X 2,000 = **0.5 sec** ➔ `loop_cnt = 0;`
- Time duty = 0.5 sec ➔ the output of "7-segment Display" is updated. (Refer to the below)
- HERA T2K Application Board : Address Mapping (0xF6XX), PORT B (7-segment Display Output)



◆ **HERA T2K Application Board (MiDAS Family)**



Lab.3 : 3) Example 2 : Code (6 of 7)

◆ **Sub Function : Initialization**

`void initialize(void)`

- Timer0 Interrupt Setting
- Initialization of Global Variable(s)

```
// Sub Function - 1. void initialize(void)
void initialize(void) {
     TMOD  = 0x22;      // Timer 0/1 : Mode 2<Auto Reload>

     TH0   = 0x06;      // TH0 for Auto-reload
                        // (Fosc/12)^-1*(256-TH0) = Timing Duty
                        // When Fosc = 12MHz and TH0 = 0x06,
                        // Timer0 interrupt interval is 0.25msec (250 usec)

     pcode = (unsigned char xdata *) 0x0000;

     ET0 = 1;           // Timer 0 Interrupt Enable
     EA =  1;           // All Interrupts Enable

     display_count = 0;
     loop_cnt = 0;

}
```

Lab.3 : 3) Example 2 : Code (7 of 7)

◆ **Interrupt Function**

```
void timer0_int(void) interrupt TF0_VECTOR
```

- Timer0 Interrupt Interval : 0.25 msec (250 usec)

```
// Interrupt Function : Timer0 Interrupt
void timer0_int(void) interrupt TF0_VECTOR {                // TF0_VECTOR : Refer to header file.

    // 7-segment Display
    if (loop_cnt == 2000) {                                 // 2,000 : 0.5sec ← 0.25msec X 2,000
        loop_cnt = 0;                                       // loop_cnt clear
        switch (display_count) {
            case 0 :
                pcode = (unsigned char xdata *) 0xF600;     // Port B (7-segment Display)
                *(pcode) = D0;                              // Refer to timer.h file about MACRO "D0".
                display_count++;
                break;
            case 1 :                 ⋮

            case 9 :
                pcode = (unsigned char xdata *) 0xF600;     // Port B (7-segment Display)
                *(pcode) = D9;                              // Refer to timer.h file about MACRO "D9".
                display_count = 0;
                break;
        }
    }
    loop_cnt++;
    TF0 = 0;                        // Timer 0 Overflow Flag Clear
}
```

Memo

Lab.4 : How to Use ADC Interrupt.

- Lab.4 : 1) Example List (Slide 1 of 9)

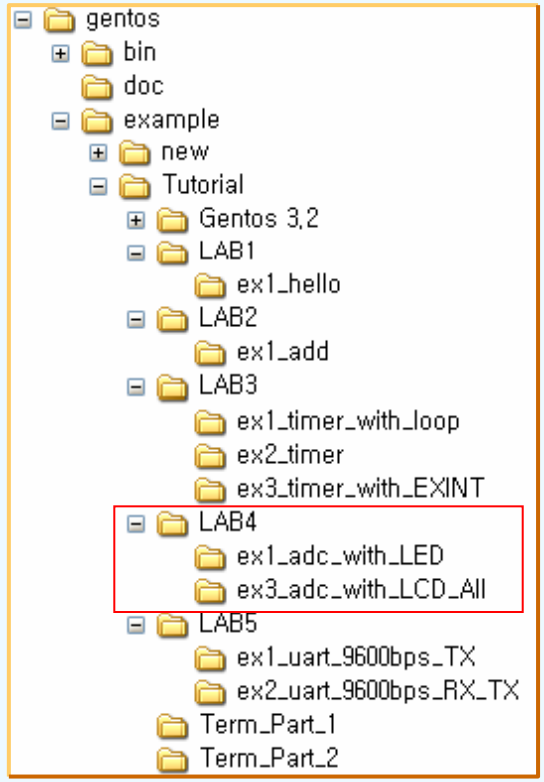
- Lab.4 : 2) Description
  - SFR : ADCSEL, ADCSEL, and ADCON (Slide 2 of 9)
  - ADC Block Diagram (Slide 3 of 9)
  - Procedure and ADC Time Chart (Slide 4 of 9)
  - ADC Interrupt Setting / ADC Run (Slide 5 of 9)

- Lab.4 : 3) Code
  - Main Function (Overview) (Slide 6 of 9)
  - Sub Function : `initialize()` (Slide 7 of 9)
  - Interrupt Function : `timer0_int()` (Slide 8 of 9)
  - Interrupt Function : `adc_int()` (Slide 9 of 9)

Lab.4 : 1) Example List (1 of 9)

- Objective : How to use ADC Interrupt
- Example List
  - ✓ ex1_adc_with_LED
  - ✓ ex2_adc_with_LCD_All
- Example : ex1_adc_with_LED
  - ✓ This example shows how to use timer0 interrupt with delay function.
  - ✓ Execution : LED (Port C) at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - Timer0 Interrupt Interval : 0.25 msec (250 usec)
    - Timer0 Mode : Mode 2 (8-bit, Auto-reload)
    - ADC Checking Interval : 0.5 sec
    - ADC Channel : ADC3 / P1.3
- Example : ex2_adc_with_LCD_All
  - ✓ This example shows how to use timer0 interrupt.
  - ✓ Execution : Character LCD Display (Port A) at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 12 MHz
    - Timer0 Interrupt Interval : 0.25 msec (250 usec)
    - Timer0 Mode : Mode 2 (8-bit, Auto-reload)
    - ADC Checking Interval : Delay Function (`for` Loop)
    - ADC Channel : ADC1 / P1.1, ADC2 / P1.2, ADC3 / P1.3

- `C:\gentos\example\Tutorial`
- Tutorial Examples is for MiDAS Family.



Lab.4 : 2) Example 1 : Description (2 of 9)

- We explain the example : ex1_adc_with_LED
- Condition
  - ✓ System Clock (F_{OSC}) : 12 MHz
  - ✓ Timer0 Interrupt Interval : 0.25 msec
  - ✓ Timer0 Mode : Mode 2
  - ✓ ADC Checking Interval : 0.5 sec
  - ✓ ADC Channel : ADC3 / P1.3

- SFR : ADCSEL

■ **ADCSEL** (E2h) : ADC Clock and Port Control Register

ADIV2	ADIV1	ADIV0	-	ADC3	ADCS2	ADCS1	ADCS0
R/W(0)	R/W(0)	R/W(0)		R/W(0)	R/W(0)	R/W(0)	R/W(0)

- ADIV2, ADIV1, ADIV0 : ADC input clock divide.
  - [0,0,0] : 1-divide (F_{OSC})
  - [0,0,1] : 2-divide (F_{OSC}/2)
  - [0,1,0] : 4-divide (F_{OSC}/4)
  - [0,1,1] : 8-divide (F_{OSC}/8)
  - [1,0,0] : 16-divide (F_{OSC}/16)
- ADC3 : 1 = ADC3 input enable & digital input disable at P1.3.
- ADC2 : 1 = ADC2 input enable & digital input disable at P1.2.
- ADC1 : 1 = ADC1 input enable & digital input disable at P1.1.
- ADC0 : 1 = ADC0 input enable & digital input disable at P0.0.

- SFR : ADCR
  - ✓ ADCR is for high 8-bit result (SAR[8:1]) of 9-bit ADC Result.
  - ✓ The other LSB 1-bit result (SAR[0]) is at ADCON.0.

■ **ADCR** (EEh) : ADC Result High Register : Value[8:1]

SAR8	SAR7	SAR6	SAR5	SAR4	SAR3	SAR2	SAR1
R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)

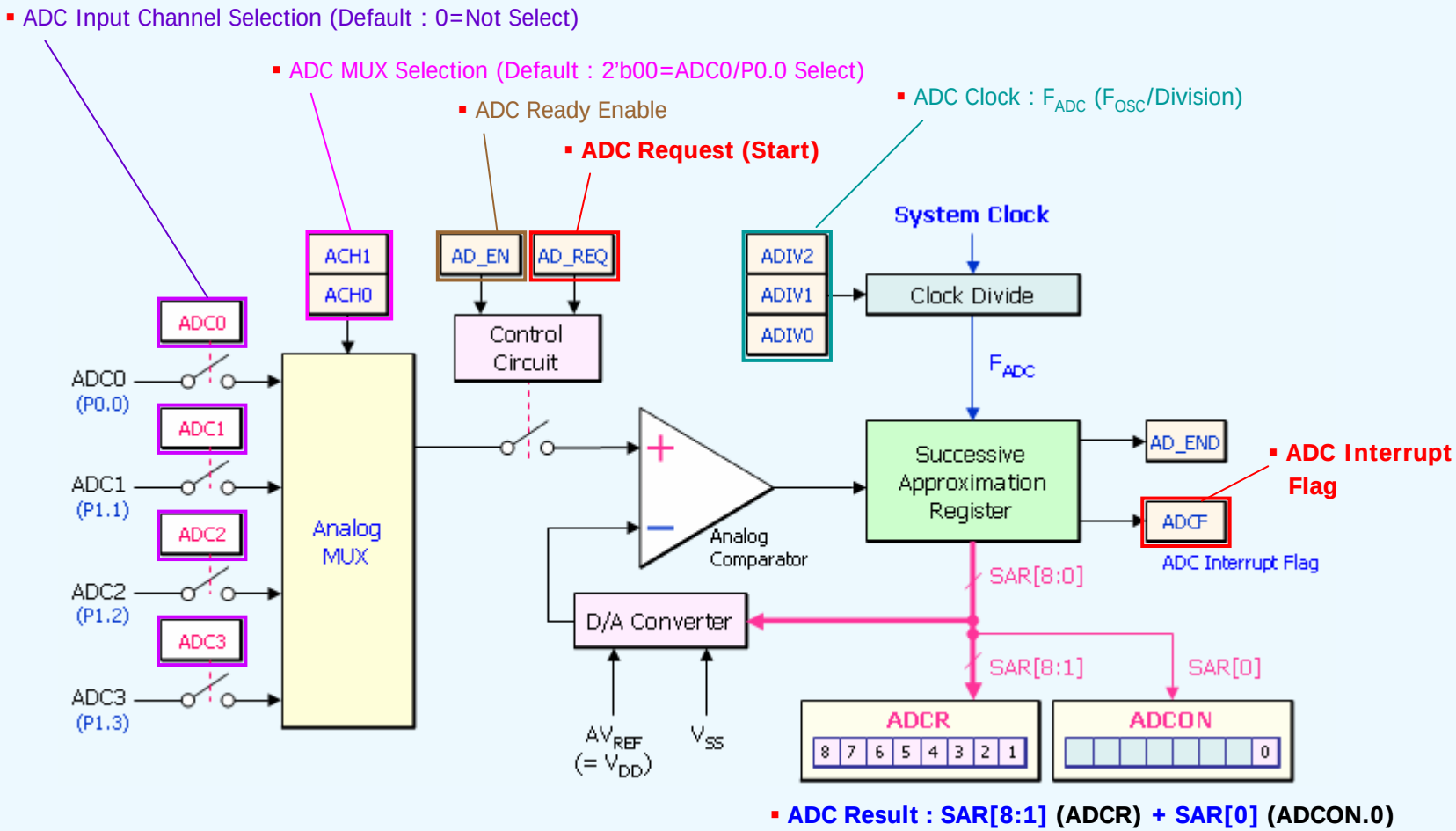
- SFR : ADCON

■ **ADCON** (EFh) : ADC Control & ADC Result LSB Register : Value[0]

AD_EN	AD_REQ	AD_END	ADCF	ACH1	ACH0	-	SAR0
R/W(0)	R/W(0)	R(1)	R/W(0)	R/W(0)	R/W(0)		R/W(0)

- AD_EN : ADC ready enable.
- AD_REQ: Request AD conversion at current channel.
  Cleared by H/W when AD_END goes to 1 from 0.
- AD_END : Current ADC status.
  0 = ADC is running now.
- ADCF : ADC interrupt flag. Must be cleared by S/W.
- ACH1, ACH0 : ADC channel selection
  - [0,0] = ADC0 input selection (P0.0)
  - [0,1] = ADC1 input selection (P1.1)
  - [1,0] = ADC2 input selection (P1.2)
  - [1,1] = ADC3 input selection (P1.3)
- SAR0 : LSB of ADC result value.

Lab.4 : 2) Example 1 : Description (3 of 9)

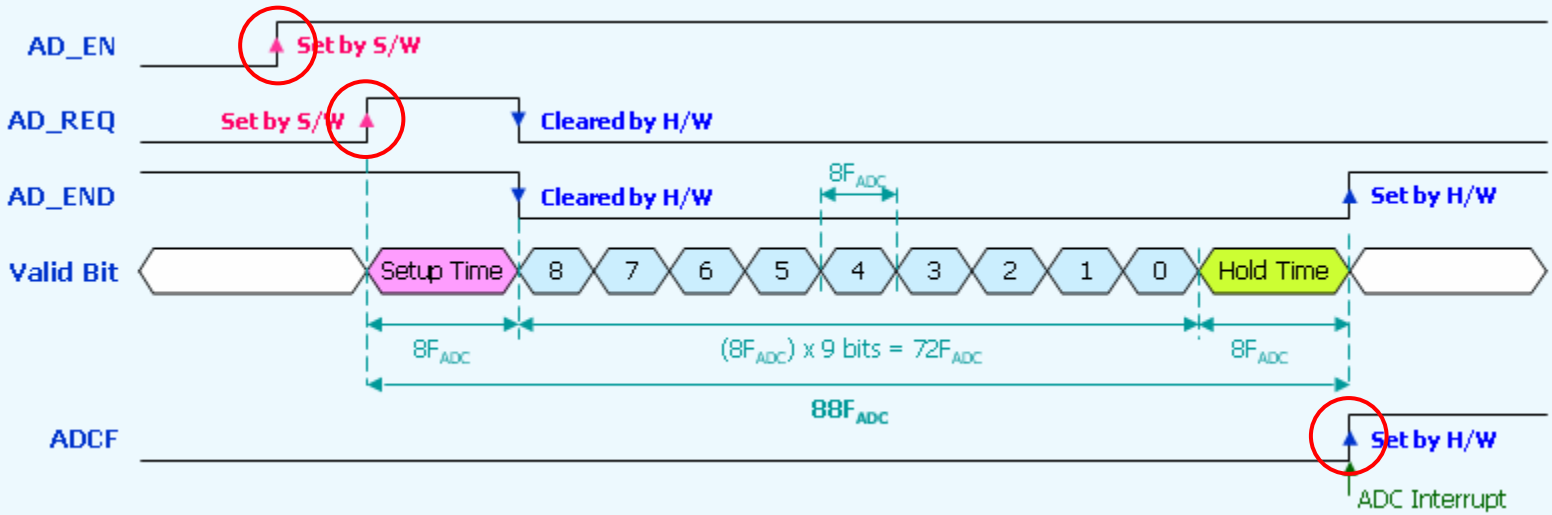


◆ **Reference** : Brief Manual of MiDAS Family (BM-MiDAS1.0-VXX.pdf) : 6-10. ADC (Analog-to-Digital Converter)

◆ **Note** : The all upper flags is "bit not accessible". Must be use with byte value for SFR : ADCSEL, ADCON, ADCR.

Lab.4 : 2) Example 1 : Description (4 of 9)

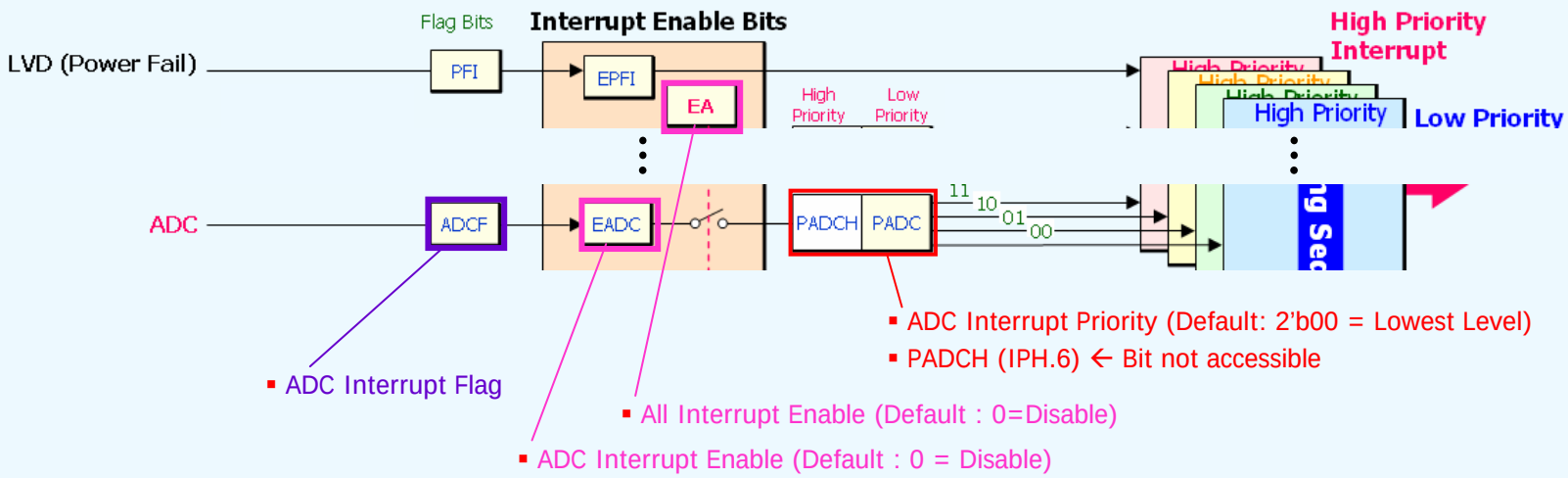
- Procedure of Using the ADC Interrupt
  - ✓ First, set the ADC input channel selection using ADC[3:0] and ADC MUX selection using ACH[1:0].
  - ✓ Set the ADC Clock (F_{ADC}) using ADIV[2:0].
  - ✓ Set the ADC Interrupt Enable and program ADC Function.
  - ✓ Set the ADC Enable (AD_EN) to "1".
  - ✓ To run the ADC function, set the ADC request (**AD_REQ**) to "1".
  - ✓ After the conversion Timing ($88F_{ADC}$), the ADC interrupt flag (**ADCF**) will be "1".
  - ✓ And ADC Function is executed. (Clear ADCF flag.)



- **Reference** : Brief Manual of MiDAS Family (BM-MiDAS1.0-VXX.pdf) : 6-11. ADC : Conversion Timing
- **Note** : If user uses the general function for ADC not interrupt function, check whether ADCF be "1" not AD_END.

Lab.4 : 2) Example 1 : Description (5 of 9)

- ADC Interrupt Setting
  - ✓ User must set the wanted interrupt enable flag to "1".
  - ✓ Because we wants to use ADC interrupt, set the "ADC interrupt enable flag (EADC)" to "1".
  - ✓ Remember that user must set the "All Interrupt Enable (EA)" to "1" to use the set interrupt(s).
  - ✓ User can set the interrupt priority.
  - ✓ Refer to below Figure and description.
- In the Example,
  - ✓ For the relative code for ADCF, refer to ADC interrupt function : `adc_int()`
  - ✓ For the relative code for EADC and EA, refer to initial setting function : `initialize()`
  - ✓ User can insert the relative code for priority flags (PADC and PADCH) to initial setting function.



- **Reference** : Brief Manual of MiDAS Family (BM-MiDAS1.0-VXX.pdf) : 6-12. Interrupt Functional Description
- **Note** : The yellow color of the upper flags is "bit-accessible", and white color is "bit not accessible".

Lab.4 : 3) Example 1 : Code (6 of 9)

◆ **Main Function**

```
// Header File : MiDAS1.0 Family
#include <GC80C520_PL44I.H>


void main(void) {

     initialize();

     TR0 = 1;       // Timer0 Run

     while(1) {
          temp = 0;
     }

}
```

◆ **Sub Function : Initialization**

```
void initialize(void)
```

- Timer0 Interrupt Setting
- ADC Interrupt Setting
- Initialization of Global Variable(s)

Timer0 Interrupt Interval : 0.25 msec

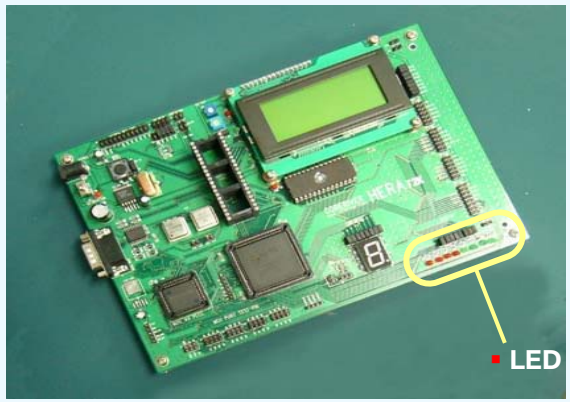
◆ **Interrupt Function**

```
void timer0_int(void) interrupt TF0_VECTOR
```

- Timer0 Interrupt Interval : 0.25 msec (250 usec)
- Global variable "loop_cnt" is incremented "+1" at each interrupt.
- `if (loop_cnt == 2,000)` ➔ 0.25 msec X 2,000 = **0.5 sec**
  ➔ `loop_cnt = 0;`
  ➔ ADC Request ➔ After $88F_{ADC}$
  ➔**ADC Interrupt Function** is run. (from `main()`)

ADC Interrupt Interval : 0.5 sec

◆ **HERA T2K Application Board (MiDAS Family)**



◆ **Interrupt Function**

```
void adc_int(void) interrupt ADC_VECTOR
```

- The output of "LED" is updated. (ADCR : SAR[8:1])
- HERA T2K Application Board : Address Mapping (0xF5XX), PORT C (LED Display Output)

Lab.4 : 3) Example 1 : Code (7 of 9)

◆ **Sub Function : Initialization**

`void initialize(void)`

- Timer0 Interrupt Setting
- ADC Interrupt Setting
- Initialization of Global Variable(s)

```
// Sub Function - 1. void initialize(void)
void initialize(void) {
     TMOD  = 0x22;      // Timer 0/1 : Mode 2<Auto Reload>

     TH0   = 0x06;      // TH0 for Auto-reload
                        // (Fosc/12)^-1*(256-TH0) = Timing Duty
                        // When Fosc = 12MHz and TH0 = 0x06,
                        // Timer0 interrupt interval is 0.25msec (250 usec)

     pcode = (unsigned char xdata *) 0x0000;

     ET0 = 1;           // Timer 0 Interrupt Enable


     ADCSEL |= 0x08;    // ADC Channel Port Selection : ADC3 (P1.3)
     ADCON  |= AD_EN_;  // ADC Enable (AD_EN_ or 0x80)
     ADCON  |= 0x0C;    // ADC Channel Analog MUX Selection : ADC3 (P1.3)

     EADC = 1;          // ADC Interrupt Enable

     EA =  1;           // All Interrupts Enable

     display_count = 0;
     loop_cnt = 0;
}
```

Lab.4 : 3) Example 1 : Code (8 of 9)

◆ **Interrupt Function**

```
void timer0_int(void) interrupt TF0_VECTOR
```

- Timer0 Interrupt Interval : 0.25 msec (250 usec)

```
// Interrupt Function : Timer0 Interrupt
void timer0_int(void) interrupt TF0_VECTOR {                // TF0_VECTOR : Refer to header file.

    // 7-segment Display
    if (loop_cnt == 2000) {                                 // 2,000 : 0.5sec ← 0.25msec X 2,000

        loop_cnt = 0;

        ADCON |= AD_REQ_;          // ADC Run/Start (AD_REQ_ or 0x40)

    }

    loop_cnt++;
    TF0 = 0;                        // Timer 0 Overflow Flag Clear
}
```

Lab.4 : 3) Example 1 : Code (9 of 9)

◆ **Interrupt Function**

```
void adc_int(void) interrupt ADC_VECTOR
```

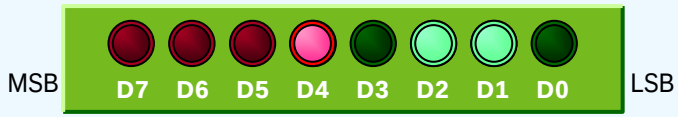
- ADC Interrupt Interval : 0.5 sec (using `timer0_int()`)

```
// Interrupt Function : ADC Interrupt
void adc_int(void) interrupt ADC_VECTOR {                    // ADC_VECTOR : Refer to header file.

     // ADC Result
     pcode = (unsigned char xdata *) 0xF500;                 // Port C (LED Display)
     *(pcode) = ADCR;                                        // ADC Result (Upper 8 bits; SAR[8:1])

     ADCON &= 0xEF;                  // ADC Interrupt Flag (ADCF) Clear
}
```

◆ **If ADC3 = 5.0V, ADCR = 0xFF (255)**

◆ **If ADC3 = 3.3V, ADCR = 0xA9 (169)**

◆ **If ADC3 = 0.5V, ADCR = 0x16 (25)**



Lab.5 : How to Use UART Interrupt.

- Lab.5 : 1) Example List (Slide 1 of 7)

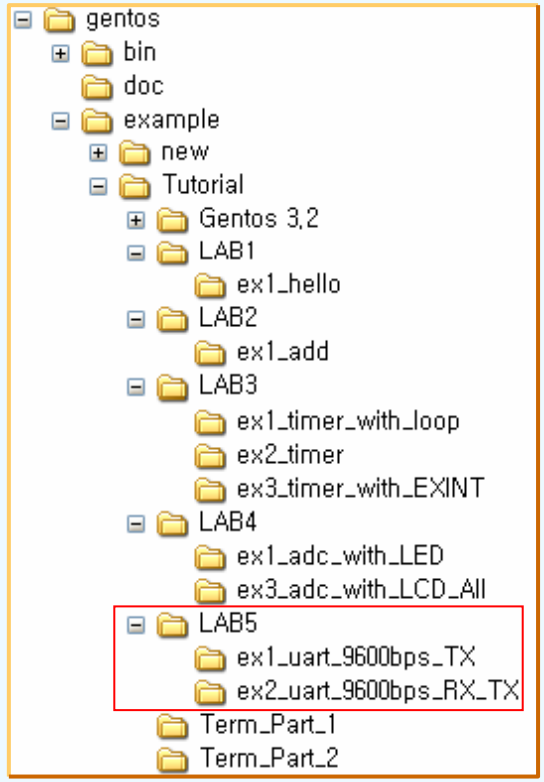
- Lab.5 : 2) Description
  - UART Mode Setting (Slide 2 of 7)
  - UART Buadrate Calculation (Slide 2 of 7)
  - SFR : PCON, CKCON, SCON, and SBUF (Slide 3 of 7)
  - UART Interrupt Setting / UART Run (Slide 4 of 7)

- Lab.5 : 3) Code
  - Main Function (Overview) (Slide 5 of 7)
  - Sub Function : `initialize()` (Slide 6 of 7)
  - Interrupt Function : `uart_int()` (Slide 7 of 7)

Lab.5 : 1) Example List (1 of 7)

- Objective : How to use UART Interrupt
- Example List
  - ✓ ex1_uart_9600bps_TX
  - ✓ ex2_uart_9600bps_RX_TX
- Example : ex1_uart_9600bps_TX
  - ✓ This example shows how to use UART interrupt.
  - ✓ Execution : 7-segment Display (Port B) and UART Transmission (TX) at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 11.0592 MHz
    - Timer0 Interrupt Interval : 0.125 msec (125 usec)
    - Timer0 & Timer1 Mode : Mode 2 (8-bit, Auto-reload)
    - UART Buadrate [bps] : 9,600 bps
    - UART Operation : Transmission at each 0.5 sec
- Example : ex2_uart_9600bps_RX_TX
  - ✓ This example shows how to use UART interrupt.
  - ✓ Execution : 7-segment Display (Port B) and UART Reception (RX) and Transmission (TX) at HERA T2K Application Board.
  - ✓ Condition
    - System Clock : 11.0592 MHz
    - Timer0 Interrupt Interval : 0.125 msec (125 usec)
    - Timer0 & Timer1 Mode : Mode 2 (8-bit, Auto-reload)
    - UART Buadrate [bps] : 9,600 bps
    - UART Operation : Reception → Transmission

- C:\gentos\example\Tutorial
- Tutorial Examples is for MiDAS Family.



Lab.5 : 2) Example 2 : Description (2 of 7)

- The objective of Lab.5 is how to use UART interrupt and function.
- We explain the example : ex2_uart_RX_TX.
- This example uses the both reception & transmission functions of the UART interrupt.
- Condition
  - ✓ System Clock (F_{OSC}) : 11.0592 MHz
  - ✓ Timer0 Interrupt Interval : 0.125 msec
  - ✓ Timer0 Mode : Mode 2
  - ✓ 7-segment Display Update : 0.5 sec
  - ✓ Timer1 Mode for UART : Mode 2
  - ✓ UART Mode : Mode 1
  - ✓ UART [bps] : 9,600 bps
  - ✓ UART Execution : RX → TX
- Mode Setting for Timer0 & Timer1
  - ✓ Refer to the "Slide 3 of 6 of Lab.3". (TMOD)

```
TMOD = 0x22;      // Timer0, Mode2
                  // Timer1, Mode2
```

- Time Calculation of Timer/Counter0.
  - ✓ Refer to the "Slide 3 of 6 of Lab.3".
- UART Mode Setting
  - ✓ SFR Register : SCON (Refer to Next Slide)
  - ✓ Mode 1 : Data Size = 10 bits
  - ✓ Start bit(1'b0) + Data(1Byte) + Stop Bit(1'b1)
- UART Buadrate Calculation for Mode 1 & 3
  - ✓ SMOD1 (PCON.7) : Default = 0. If 1 → Double Baudrate.
  - ✓ T1M (CKCON.4) : Default = 0. (F_{OSC}/12). If 1 → F_{OSC}/4

$$\text{Buadrate[bps]} = \frac{2^{SMOD1}}{32} \times \underbrace{F_{OSC} \times \frac{3^{T1M}}{12} \times \frac{1}{256-(TH1)}}_{\text{Timer1 Overflow}}$$

- SMOD1 & T1M = 0 (Default)
- System Clock F_{OSC} = 11.0592 MHz
- TH1 = 0xFD (253)

$$\frac{1}{32} \times 11.0592 \times 10^6 \times \frac{1}{12} \times \frac{1}{256-(253)} = 9{,}600\,[\text{bps}]$$

Lab.5 : 2) Example 2 : Description (3 of 7)

SFR : PCON

PCON (87h) : Power Control Register

SMOD1	SMOD0	-	POF	GF1	GF0	PD	IDL
R/W(0)	R/W(0)		R/W(1)	R/W(0)	R/W(0)	R/W(0)	R/W(0)

- SMOD1 : Timer 1 baudrate double in UART mode 1, 2, 3.
- SMOD0 : Enable SM0 access. Don't modify this bit.
- POF : Power off flag.
  When power-on, this bit will be set by H/W.
- GF1, GF0 : General purpose flag bit.
- PD : Power-down (Stop) mode enable.
- IDL : IDL mode enable.

SFR : CKCON

CKCON (8Eh) : Clock Control Register

WD1	WD0	T2M	T1M	T0M	-	-	-
R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)			

- WD1, WD0 : Watchdog timer mode select
  [0,0] : 2^{17} clocks (interrupt), 2^{17} + 512 clocks (reset)
  [0,1] : 2^{20} clocks (interrupt), 2^{20} + 512 clocks (reset)
  [1,0] : 2^{23} clocks (interrupt), 2^{23} + 512 clocks (reset)
  [1,1] : 2^{26} clocks (interrupt), 2^{26} + 512 clocks (reset)
- T2M : Timer 2 clock select. When set, base time is 4 clocks.
- T1M : Timer 1 clock select. When set, base time is 4 clocks.
- T0M : Timer 0 clock select. When set, base time is 4 clocks.

SFR : SCON

SCON (98h) : Serial Port Control Register of UART0

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)

- SM0, SM1 : Serial Port mode select.
  [0,0] : Mode0, 8-bit shift register (F_{OSC}/4)
  [0,1] : Mode1, 8-bit UART (Variable)
  [1,0] : Mode2, 9-bit UART (F_{OSC}/32 or F_{OSC}/16)
  [1,1] : Mode3, 9-bit UART (Variable)
- SM2 : Enables the Automatic Address Recognition in Mode2 and 3.
  In Mode1, valid stop bit check if SM2 is "1".
  In Mode0, SM2 should be "0".
- REN : Serial reception enable. (P3.0 I/O pin ➔ RXD pin)
- TB8 : 9th data bit that will be transmitted in Mode2 and 3.
- RB8 : 9th data bit that was received in Mode 2 and 3.
  In Mode1, RB8 is equal to stop bit if SM2 is "0".
  In Mode0, RB8 is not used.
- TI : Transmission interrupt flag. Must be cleared by S/W.
- RI : Reception interrupt flag. Must be cleared by S/W.

SFR : SBUF

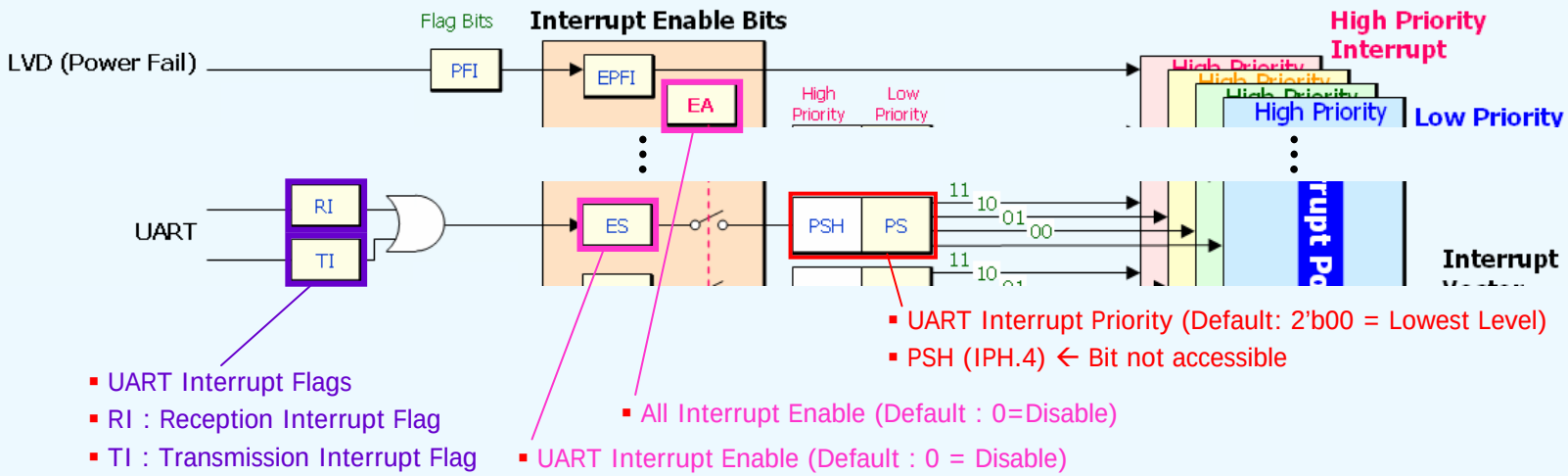
SBUF (99h) : Serial Data Buffer Register

SBUF.7	SBUF.6	SBUF.5	SBUF.4	SBUF.3	SBUF.2	SBUF.1	SBUF.0
R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)	R/W(0)

- Transmission buffer and reception buffer are separated.
- Read and write address are same.

Lab.5 : 2) Example 2 : Description (4 of 7)

- UART Interrupt Setting
  - ✓ User must set the wanted interrupt enable flag to "1".
  - ✓ Because we wants to use UART interrupt, set the "UART interrupt enable flag (ES)" to "1".
  - ✓ Remember that user must set the "All Interrupt Enable (EA)" to "1" to use the set interrupt(s).
  - ✓ User can set the interrupt priority.
  - ✓ Refer to below Figure and description.
- In the Example,
  - ✓ For the relative code for RI and TI, refer to UART interrupt function : `uart_int()`
  - ✓ For the relative code for ES and EA, refer to initial setting function : `initialize()`
  - ✓ User can insert the relative code for priority flags (PS and PSH) to initial setting function.
  - ✓ For Timer1 (UART) run, user must insert the "TR1 = 1".



◆ **Reference** : Brief Manual of MiDAS Family (BM-MiDAS1.0-VXX.pdf) : 6-12. Interrupt Functional Description

◆ **Note** : The yellow color of the upper flags is "bit-accessible", and white color is "bit not accessible".

Lab.5 : 3) Example 2 : Code (5 of 7)

◆ Main Function

```
// Header File : MiDAS1.0 Family
#include <GC80C520_PL44I.H>


void main(void) {

     initialize();

     TR0 = 1;       // Timer0 Run
     TR1 = 1;       // Timer1 Run
                    // UART Run
     while(1) {
          temp = 0;
     }
}
```

◆ Sub Function : Initialization

```
void initialize(void)
```

- Timer0 Interrupt Setting
- UART Interrupt Setting (Including Timer1 Setting)
- Initialization of Global Variable(s)

Timer0 Interrupt Interval : 0.125 msec

◆ Interrupt Function

```
void timer0_int(void) interrupt TF0_VECTOR
```

- Timer0 Interrupt Interval : 0.125 msec (125 usec)
- Global variable “loop_cnt” is incremented “+1” at each interrupt.
- `if (loop_cnt == 4000)` ➔ 0.25 msec X 4,000 = **0.5 sec** ➔ `loop_cnt = 0;`
- Time duty = 0.5 sec ➔ the output of “7-segment Display” is updated. (Refer to the slide 5 & 7 of 6 of Lab.3 about the sub function)

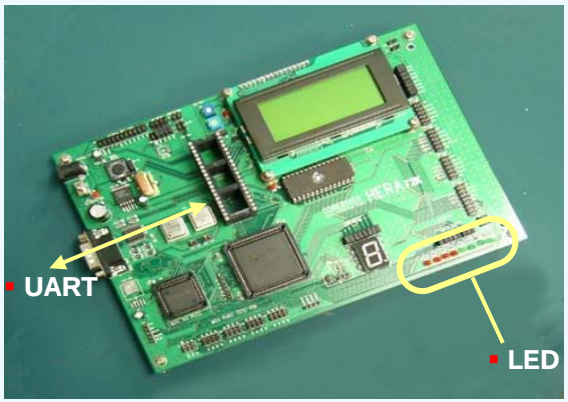
UART Interrupt When RX or TX is occurred

◆ Interrupt Function

```
void uart_int(void) interrupt SIO_VECTOR
```

- When the data through serial port to RX pin of MCU is received, UART interrupt is occurred (RI = 1).
- And the received data is transmitted using SBUF register, then UART interrupt is occurred (TI = 1).
- This UART interrupt function is programmed for the Reception & Transmission Operation.

◆ HERA T2K Application Board (MiDAS Family)



Lab.5 : 3) Example 2 : Code (6 of 7)

◆ **Sub Function : Initialization**

`void initialize(void)`

- Timer0 Interrupt Setting
- ADC Interrupt Setting
- Initialization of Global Variable(s)

```
// Sub Function - 1. void initialize(void)
void initialize(void) {
     TMOD  = 0x22;      // Timer 0/1 : Mode 2<Auto Reload>

     TH0   = 0x78;      // TH0 for Auto-reload
                        // Timer0 interrupt interval is 0.125msec (125 usec)

     ET0 = 1;           // Timer 0 Interrupt Enable

     SM0 = 0; SM1 = 1;   // UART Mode 1 (8-bit UART) : SM0,SM1 =[0,1]

     TH1 = 0xFD;        // Timer 1 Setting for UART --> 9,600 bps

     REN = 1;           // Reception Enable : RXD (P3.0) is enabled.
                        // If user REN is not set to "1", the data can not be received.

     ES = 1;            // UART Interrupt Enable
     PS = 1;            // UART Interrupt Priority Enable

     EA =  1;           // All Interrupts Enable

     pcode = (unsigned char xdata *) 0x0000;
     display_count = 0;
     loop_cnt = 0;
}
```

```
// If user want the double baudrate setting,
// insert the below code. (9,600 --> 18,200)
PCON  |= 0x80;
// or
PCON  |= SMOD1_;
```

Lab.5 : 3) Example 2 : Code (7 of 7)

◆ **Interrupt Function**

```
void uart_int(void) interrupt SIO_VECTOR
```

- When the data reception using RX pin of MCU or transmission using TX pin is occurred with SBUF, UART interrupt is occurred.

```
// Interrupt Function : UART Interrupt (Serial Communication)
void uart_int(void) interrupt SIO_VECTOR {      // SIO_VECTOR : Refer to header file.

    while (1) {

        if (TI) {                   // [Transmit Mode]

            TI = 0;                 // Transmission Interrupt Flag Clear
            break;

        } else if (RI) {            // [Receive Mode]

            received_data = SBUF;   // Read the received data from SBUF (SBUF <-- RX pin)
            SBUF = received_data;   // Resend the data to TX pin using SBUF
                                    //   After the reception interrupt is finished,
                                    //   the transmission interrupt is occurred and resend the data.

            RI = 0;                 // Reception Interrupt Flag Clear
            break;

        }
    }
}
```