|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | ***Portland State University*** [Maseeh College of Engineering and Computer Science](http://pdx.edu/cecs/) 1559170_300.jpg | | ECE 372 - PROJECT REPORT | | Project 1: TIMER | | **1/20/2015** | |
|  |
| **Instructor: Douglas V.Hall**  **TA: Leela Yadlapalli** |
| Students: Hai Dang Hoang  PSUID: 946749560  Email: [danghai@pdx.edu](mailto:danghai@pdx.edu) |

**Introduction:**

In Design Project #2 in ECE 371 I developed a program that created a rotating LED display on the LEDs and a program that turned on an LED when a debounced button was pushed once and turned the LED off when the button was pushed again. For timing the rotating LED display I used a delay loop. For this exercise I will extend this project so that the rotating LED display is turned on and off with button pushes and the on and off times for the LEDs are determined by a programmable timer.

As before, I will be using the 4 User LEDs on the BeagleBone Black boards and the output from a debounced, pushbutton switch that has been connected to GPIO1\_31 on each board. I will use Timer5 of the AM3359 processor to turn each LED on for 1 second and off for 1 second.

1. **GPIO Table:**

In this program, I will use GPIO1\_21, GPIO1\_22, GPIO1\_23, GPIO1\_24 ( LED on the BeagleBone Black board) to rotate.

**Table 1: Template for initializing GPIO1\_21-24 for SetDataOut and ClearDataOut**

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

Hex: 0x01E00000

**Table 2: Template for initializing GPIO1\_21-24 as output**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **GPIO** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** |
| **Bit** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| **HEX** | F | | | | E | | | | 1 | | | |
| **GPIO** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| **Bit** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **HEX** | F | | | | F | | | | F | | | |
| **GPIO** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Bit** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **HEX** | F | | | | F | | | |

Hex: 0xFE1FFFFF

**Table 3: Template for turning on USER LED 0 only**

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

Hex: 0x00200000

**Table 4: Word for turn on USER LED 1 only**

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

Hex: 0x00400000

**Table 5: Word for turn on USER LED 2 only**

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

Hex: 0x00800000

**Table 6: Word for turn on USER LED 3 only**

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

Hex: 0x01000000

1. **Standard Program Structure and Algorithm:**
2. **Mainline**

* INITIALIZE STACK
* INITIALIZE GPIO1

Initial base address for GPIO1 (0x4804C000)

Set GPIO1 bits 21-24 to low by write 0x01E00000 to GPIO1\_CLEARDATAOUT at 0x4804C190 (0x4804C000 + 0x190 offset for GPIO\_CLEARDATAOUT)

Set GPIO1 bits 21-24 to output by R-M-W 0xFE1FFFFF to GPIO1\_OE at 0x4804C134 (0x4804C000 + 0x134 offset for GPIO\_OE)

Set GPIO1\_31 to detect falling edge by RMW 0x80000000 to GPIO1\_FALLINGDETECT at 0x4804C14C (0x4804C000 + 0x14C offset for GPIO\_FALLINGDETECT)

Enable GPIO\_31 request on GPIO1\_IRQSTATUS\_SET\_0 by write 0x80000000 to 0x4804C034

* INITIALIZE INT

Reset the software by writing 0x2 to 0x48200010 (0x48200000( INTC\_Base) + 0x10 (offset of INT\_SYSCONFIG))

Enable TIMER2 interrupt by writing 0x10 to 0x482000C8 (0x10 for unmasking INT 68, 0x482000C8 = 0x48200000 (INTC\_Base) + 0xC8 ( Offset of INTC\_MIR\_CLEAR2).

Enable GPIO\_1 interrupt by enable Int number 98 of the INTC (write 0x04 to 0x482000E8: 0x48200000 (base address for INTC) + 0xE8 offset for INTC\_MIR\_CLEAR3).

* INITIALIZATION TIMER2

Enable clock for TIMER2 by writing 0x2 to 0x44E00080 ( CM\_PER\_TIMER2\_CLKCTRL)

Reset the software by storing 0x1 to 0x48040010 ( 0x48040000 (Timer2\_base + 0x10 (offset of Timer OCP configuration)).

Setting value for TCRR by storing 0xFFFF8300 to 0x4804003C ( 0x48040000 (Timer2\_base + 0x3C (offset of Timer Counter Register)).

Setting value for TLDR by storing 0xFFFF8300 to 0x48040040 (0x48040000 (Timer2\_base) + 0x40 (offset of Timer Load Register)).

Enable auto reload by writing 0x3 to 0x48040038 (0x3 for enable Auto-reload timer and start the timer, 0x48040038 =0x48040000 (Timer2\_base) + 0x38 (offset of Timer Control Register)).

Enable IRQ overflow interrupt for TIMER 2 by writing 0x2 to 0x4804002C (0x2 to enable IRQ for Overflow, 0x4804002C = 0x48040000 (Timer2\_base) + 0x2C (offset of Timer Interrupt Enable Set Register)).

* INITIALIZE a register equal 0 ( Register control index to get value from pointer of status 4 LED from Rotate\_LED)
* ENABLE the Processor IRQ

Copy current value in CPSR into a register

Clear bit 7 of the current CPSR

Write the modified result back to CPSR ( 8 bit lowest).

* LOOP forever ( Endless loop) : Wait for interrupt signal.

1. **Interrupt Procedure (INT\_DIRECTOR)**

* Saved uses register and linked register on Stack
* Check if the interrupt come from GPIO1 by test bit 2 (int number 98) of the current value stored in INTC\_PENDING\_IRQ3 (0x482000F8: 0x48200000 (base address for INTC) + 0xF8 offset for INTC\_PENDING\_IRQ3)
* If the interrupt not come from GPIO\_1 ( bit 2 != 0)

Go to Check\_Check\_TIMER\_interrupt procedure

* Else

Check if the interrupt comes from GPIO1 bit 31 by test bit 31 (GPIO1\_31) of the current value store in GPIO1\_IRQSTATUS\_0 (0x4804C02C: 0x4804C000 + 0x2C offset for GPIO\_IRQSTATUS\_0).

If the interrupt come from GPIO\_1 bit 31 ( Bit 31 ==1)

Go to BUTTON\_SVC

Else ( Go to PASS\_ON)

Restore saved register

Return from IRQ interrupt procedure

1. **Button Service Procedure :**

* Turn off IRQ request from GPIO\_31 by writing 0x80000000 to 0x4804C02C (GPIO1\_IRQ\_STATUS\_0).
* Generate new IRQ by writing 0x1 to 0x48200048 ( INTC\_CONTROL).
* Turn off all 4 USER LEDS by writing 0x01E00000 into 0x4804C190 (0x4804C000 (GPIO1\_Base) + 0x190 (offset of GPIO\_CLEAR\_DATAOUT)) to avoid error.
* Check Flag\_button ( If flag\_button == 0)

{

Update flag =1;

Reset Flag\_timer back to 0;

Reset pointer to control Rotating LED to original;

Go to PASS\_ON

}

Else

{

Update flag = 0

Go to PASS\_ON

}

1. **Check\_TIMER\_interrupt procedure:**

* Read INTC\_PENDING\_IRQ2 Register at 0x482000D8.
* Check TIMER overflow interrupt ( if bit 4 == 0) 🡪 No timer overflow interrupt and continuous counting

Go to PASS\_ON

Else

Go to Timer\_IRQ procedure.

1. **Timer\_IRQ procedure:**

* Turn off IRQ request from overflow by writing 0x2 to to 0x48040028 (0x48040000 (Timer2\_base) + 0x28 (offset of Timer\_IRQ STATUS)).
* Generate new IRQ generation by writing 0x1 to 0x48200048 ( INTC\_CONTROL).
* Check Flag\_Button ( if Flag\_button ==0)

{

Turn off all 4 USER LEDS by writing 0x01E00000 into 0x4804C190 (0x4804C000 (GPIO1\_Base) + 0x190 (offset of GPIO\_CLEAR\_DATAOUT)).

Go to PASS\_ON

}

Else

{

Check Flag\_timer ( if Flag\_timer == 0)

{

Turn on LED depending on R5 pointer.

Update pointer R5 ( If R5 == 0xC 🡪 Restore to 0x0, else Increment pointer by 4 to get the next value in the Rotate\_LED).

Update Flag\_timer =1

}

Else

{

Turn off all LED by writing 0x01E00000 into 0x4804C190.

Update Flag\_timer = 0

Go to PASS\_ON

}

}

1. **PASS\_ON:**

* Load register from Stack
* Go back to mainline

1. **Initialize Data:**

* Flag\_button: .word 0x0.

Flag\_button = 1 🡪 Rotate\_LED turn on

Flag\_button = 0 🡪 Rotate\_LED turn off

* Flag\_timer: .word 0x1

Flag\_timer = 0 🡪 Time (1 second) for LED off

Flag\_timer = 1 🡪 Timer (1 second) for LED on

* Rotate\_LED .word 0x01000000, 0x00800000, 0x00400000, 0x00200000

**Signed Statement**

I developed and wrote this program by myself with no help from anyone except the instructor and the T.A. and I did not give help to anyone else