IS1200 Lab 3

Questions for Assignment 1

Test pressing BTN3 and BTN2 at the same time. What happens? Why?

The right minute digit and left second digit are changed to whatever hexadecimal number the switches correspond to. *Explain why the code works*.

Three device-registers for input/output control are TRISE, TRISESET, and TRISECLR. Their functions are related. How? What are the differences?

TRISE, TRISESET and TRISECLR are all used to configure the mask of LEDs, which is stored in TRISE.

The bits in TRISE represent the mask that decide which LED can be turned ON or OFF.

The bits in TRISESET are used to turn specific bits in TRISE ON.

The bits in TRISECLR are used to turn specific bits in TRISE OFF.

In the generated assembly code, in which MIPS register will the return values from functions getbtns and getsw be placed in. You should be able to answer this question without debugging the generated assembly code.

$v0, because the return value is contained in 32 bits

In this exercise, we explained which bits that should be used in Port D and Port E. How can you find this information in the PIC32 and ChipKIT manuals? Be prepared to demonstrate how to find this information in the manuals.

Demonstrate how to find this information

Questions for Assignment 2

When the time-out event-flag is a "1", how does your code reset it to "0"?

```
IFSCLR(0) = 0x100;
```

What would happen if the time-out event-flag was not reset to "0" by your code? Why?

It would run as fast as possible since no new interrupt would happen after the first one.

Which device-register (or registers) must be written to define the time between time-out events? Describe the function of that register (or of those registers).

T2CON to set prescaling. PR2 to define the period.

If you press BTN3 quickly, does the time update reliably? Why, or why not? If not, would that be easy to change? If so, how?

The time only updates if BTN3 is pressed when a tick happens.

Questions for Assignment 3

When the time-out event-flag is a "1", how does your code reset it to "0"?

```
IFSCLR(0) = 0x100;
```

What would happen if the time-out event-flag was not reset to "0" by your code? Why?

It would run as fast as possible since no new interrupt would happen after the first one.

From which part of the code is the function user_isr called? Why is it called from there?

It is called in vectors.S since interrupts need to be called from there.

Why are registers saved before the call to user_isr? Why are only some registers saved?

Because whenever an interrupt occurs the rest of the code is put on hold to execute an interrupt service routine and this might make use of registers which are used in the main part of the code, so therefore we save all the registers to the stack (but no $Sx registers since they are not caller-save registers).

> The registers that are saved are of the non-preserved type.
> Since we have a function _isr_trampoline (caller) that calls user_isr (callee), it is the callers responsibility, according to to general practice, to save the non-preserved registers in case it wants to use them after the function call.
>
> *"Registers $1 through $15, $24, and $25 can be changed by any C function, or any assembler subroutine. Register $ra will be changed by the call to the C function, i.e., the jal instruction. Saving these registers is necessary before calling the C function user_isr."*

Which device-register (or registers), and which processor-register (or registers) must be written to enable interrupts from the timer? Describe the functions of the relevant registers.

Device-registers: IEC0 to enable interrupts. IPC2 to configure priority.

Processor-registers: ???