**Reading Assignment**

- Software and Hardware Engineering: Chapter 3, 8, 9 (New version)
  
  Or

- Microcontrollers and Microcomputers: Chapter 6

- Software and Hardware Engineering: Chapter 6 (Old version)

---

**Software Design**

- Means designing the software before writing the program code
- The general approach is to learn the instruction set and the syntax first, without too much design
- As you become familiar with the processor, work on designing the solution, rather than just coding the solution
- Designing the software is more than just writing the software!

---

**Use Flowcharts to Plan Program Structure**

Flow Chart Symbols

- START
- LABEL:
- OPERATION
- Y (YES)
- N (NO)
- CONDITIONAL BRANCH
- END
**IF-THEN Flow Structure**

```
if (C)
  { A;
  }
```

---

**Example: IF-THEN**

**Pseudocode**

```
if (A<10)
  { var = 5;
  }
```

**Assembly Code**

```
CMPA #10
BLT L1
L1: LDAB #5
STAB var
L2: next instruction
```

---

**IF-THEN-ELSE Flow Structure**

```
if (C)
  { A;
  }
else
  { B;
  }
```

---

**Example: IF-THEN-ELSE**

**Pseudocode**

```
if (A<10)
  { var = 5;
  }
else
  { var = 0;
  }
```

**Assembly Code**

```
CMPA #10
BLO L1
L1: LDAB #5
STAB var
L2: next instruction
```
Example: IF-THEN-ELSE Decision

; Get Temperature
ldaa AD_PORT
; IF Temperature > Allowed Maximum
cmpa #MAX_ALLOWED
bpl ELSE_PART ; branch lower or same (u.b.)
; THEN Turn the water valve off
ldaa VALVE_OFF
staa VALVE_PORT
bra END_IF
; ELSE Turn the water valve on
ELSE_PART:
ldaa VALVE_ON
staa VALVE_PORT
END_IF:

DO-WHILE Flow Structure

do
{  A;
} while (C);

Example: DO-WHILE

Pseudocode

$i = 0;$
do
{  table[$i$] = table[$i/2$];
    $i = i+1$;
} while ($i <= LEN$);

Example: DO-WHILE Repetition

; DO
DO_BEGIN:
; Get data from the switches
ldaa SW_PORT
; Output the data to the LEDs
staa LEDs
; END_DO
; WHILE Any switch is set
bt SW_PORT
bne DO_BEGIN
; END_WHILE
### WHILE Flow Structure

```plaintext
while (C)
{
  A;
}
```

### Example: WHILE

**Pseudocode**

```plaintext
i = 0;
while (i < LEN)
{
  table[i] = table[i]*2;
  i = i + 1;
}
```

**Assembly Code**

```plaintext
LDX #table
CLRA
L1: CMPA #LEN
BLT L2
BRA L3
L2: ASL X,#1
INCA BRA L1
L3: next instruction
bra WHILE_START
END_WHILE:
```

### Example: WHILE-DO Repetition

- Get the temperature from the A/D
- lda AD_PORT
- WHILE the temperature >= maximum allowed
  - WHILE_START:
    - cmpa MAX_ALLOWED
    - bra END_WHILE
  - DO
    - ; work inside the loop
    - lda AD_PORT
  - END_DO
- bra WHILE_START
- END_WHILE:

### Top-Down Design: An Example

- **Problem:** Start with a table of data. The table consists of 5 values. Each value is between 0 and 255. Create a new table whose contents are the original table divided by 2.
Top-Down Design (1)

- Step 1: Determine where code and data will go in memory
  - E.g. Code at $1000, data at $2000
- Step 2: Determine type of variables to use
  - Because data will be between 0 and 255, can use unsigned 8-bit numbers
- Step 3: Draw a picture of the data structures in memory

Top-Down Design (2)

- Strategy: Because we are using a table of data, we will need pointers to each table so we can keep track of which table element we are working on
  - use the X and Y registers as pointers to the tables

Top-Down Design (3)

- Step 4: Use a simple flow chart to plan structure of program

Top-Down Design (4)

- Need a way to determine when we reach the end of the table
  - One way: Use a counter (say, register A) to keep track of how many elements we have processed
Top-Down Design (5)

- Step 5: Add code to implement blocks

Top-Down Design (6)

- Step 6: Write program

Summary of Top-Down Program Design

- Plan structures in memory
- Start with a large picture of program structure
- Work down to more detailed structure
- Translate structure into code
- Optimize for efficiency