Stockholm University

Introduction to Datorsystem labs, VT2022

Ramin Firouzi
Mahbub Ul Alam

Overview

Stockholm University

- Introduction to Assembly programming language
- Machine language
- Learn basics of MOP (Machine-Oriented Programming)
- How assembly is translated into machine code
- Flowcharts and subroutines
- Introduction to CPUlator Simulator and DE2 board
- Lab 1 overview
- Lab 2 overview

Assembly Language

- Assembly language is just a more readable version of machine language

Levels of programming languages

Stockholm University

- Machine Language
- Assembly language (low-level language)
- High Level Languages

Machine Language

Stockholm University

- The only language that is ever actually executed by a computer.
- Composed of instructions encoded as strings of 0's and 1's.
- Never written and rarely read directly by programmers.
  - Not user friendly; sequence of 0s and 1s

Assembly Language (Low Level Language)

- Machine oriented
- Not portable
- Designed for a specific family of processors (different processor groups/family has different Assembly Language)
- Instruction sets are represented by ***mnemonics*** i.e. set of letters
- For example: movia, bne, sub, add

High Level Languages

Stockholm University

- e.g. : C, C++, JAVA and Vbasic
- Designed to eliminate the technicalities of a particular computer.
- Portable and machine independent
- Human friendly
- Problem-oriented

Advantages of Assembly language

Stockholm University

- Faster
- Requires less memory
- Sometimes an operation can only be performed by using Assembly

Basic MOP (Machine-Oriented Programming)

Stockholm University

add

Operation:	rC ← rA + rB
Assembler Syntax:	add rC, rA, rB
Example:	add r6, r7, r8
Description:	Calculates the sum of rA and rB. Stores the result in rC. Used for both signed and unsigned addition.

Basic MOP (Machine-Oriented Programming)

Beq (branch if equal)

Operation:	if (rA == rB) then PC ← PC + 4 + σ (IMM16) else PC ← PC + 4
Assembler Syntax:	beq rA, rB, label
Example:	beq r6, r7, label
Description:	If rA == rB, then beq transfers program control to the instruction at label.

Basic MOP (Machine-Oriented Programming)

Stockholm University

movia (move immediate address into word)

Operation:	rB ← label
Assembler Syntax:	movia rB, label
Example:	movia r6, function_address
Description:	Writes the address of label to rB.

Basic MOP (Machine-Oriented Programming)

Stockholm University

subi (subtract immediate)

Operation:	rB ← rA – σ (IMMED)
Assembler Syntax:	subi rB, rA, IMMED Example: subi r8, r8, 4
Example:	subi r8, r8, 4
Description:	Sign-extends the immediate value IMMED to 32 bits, subtracts it from the value of rA and then stores the result in rB.

Stockholm University

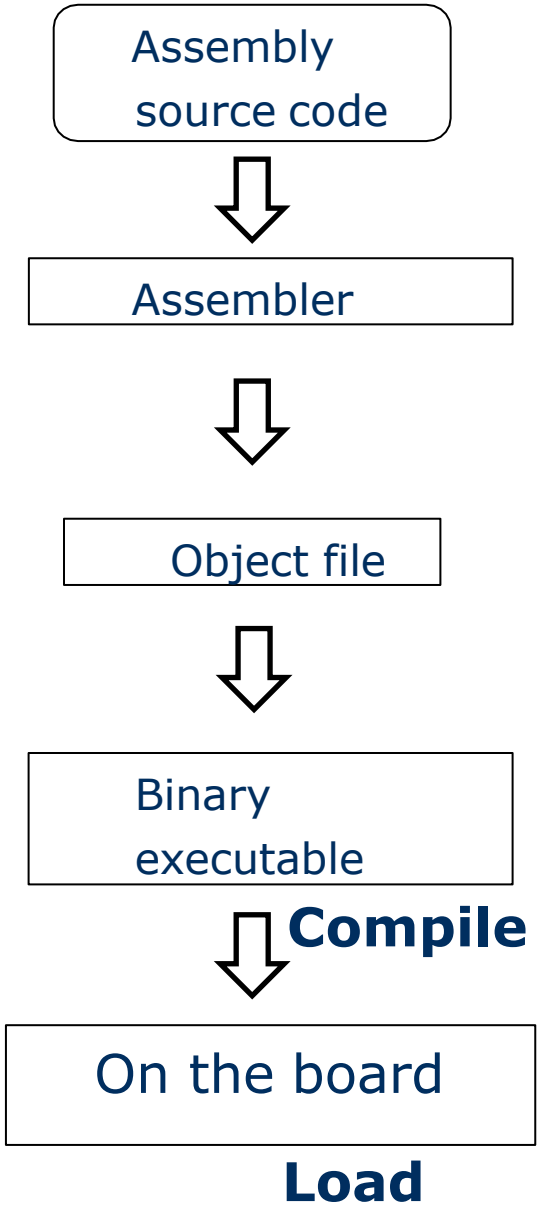
Program Components

- Directives give information to the assembler and are not translated
- Lables are used to refer the location by name instead of address
- Instructions are executed by the assembler
- Comments are for documenting your code

Stockholm University

Subroutine

- A subroutine in a self-contained section of code that implements a specific function that can be called from many different places.
  - Save memory
  - Improve reusability of code
  - Better organization



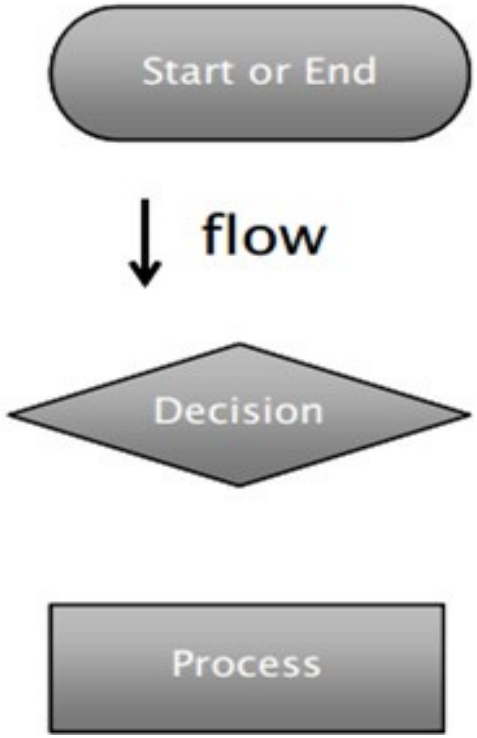
The assembler parses your source code, checks it for syntax errors, and translates it into its equivalent *binary* form i.e. object file

The *linker* pastes all the object files together grouping their data and code sections to form an executable program

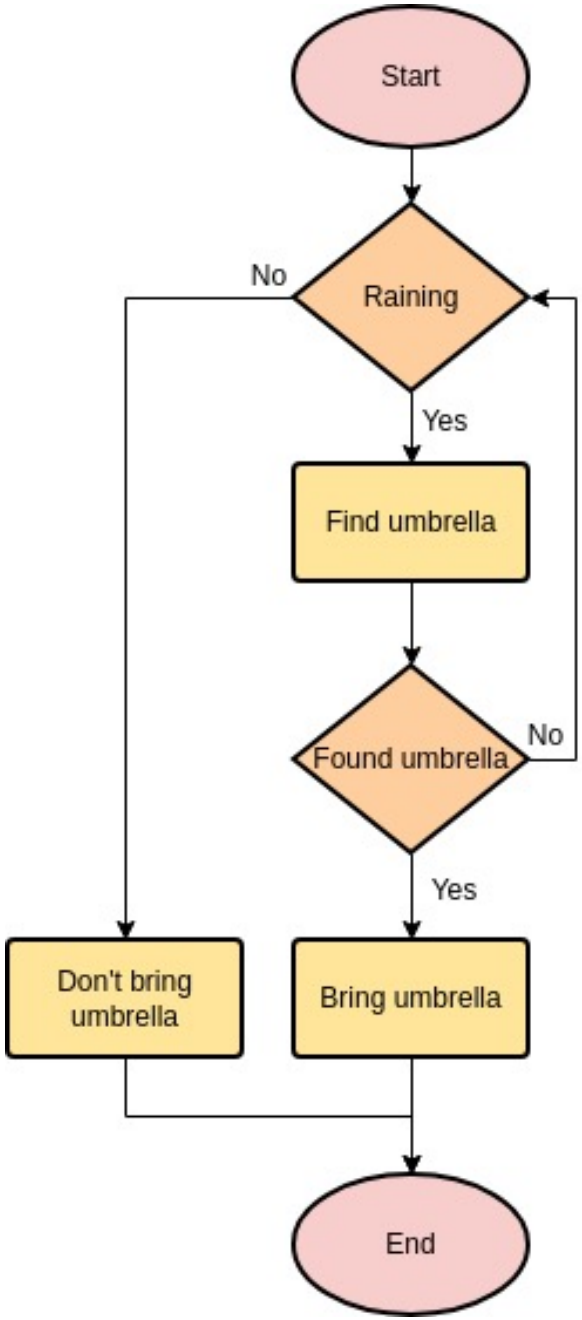
The executable file contains data and instructions that are encoded specially for Nios II processor

Flowcharts

Meaning of symbols



Flowchart example



Stockholm University

After the lab # 1

- able to understand how flow charts work
- be able to write simple assembly code based on flow charts and manifestos
- understanding of how the Assembly is translated into machine
- practical experience of how the simulator can be used.

Stockholm University

Before the lab

- Read through the entire lab manual
- Solved all preparation tasks on the course website successfully
- Read Installation manual of simulators

Stockholm University

During lab

- We will use simulator
- Two version of the simulator is available— web based and application, which can be installed on computer
- It is simulator for NIOS/IIs processor

Stockholm University

During lab

- Study the sample program and run it
- Make some changes and see what happens
- Document the code
- And, then create your own Assemble code based on the given flow-charts.
- Document the code

Stockholm University

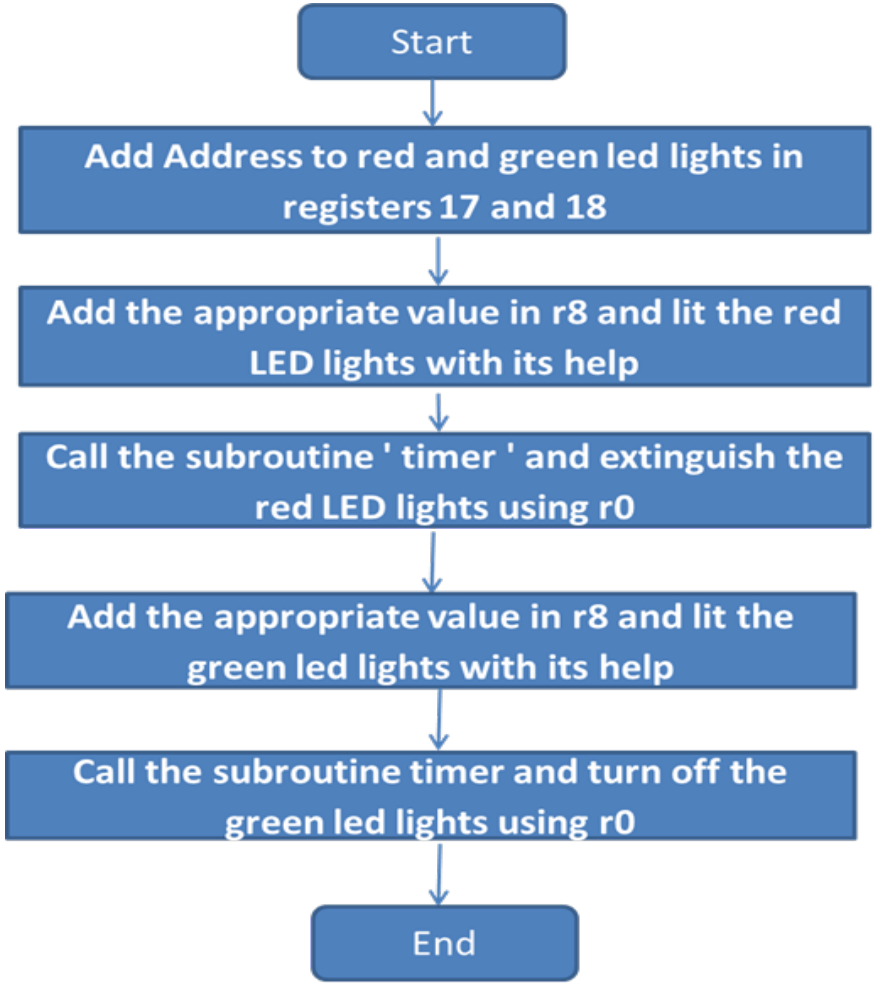
Example

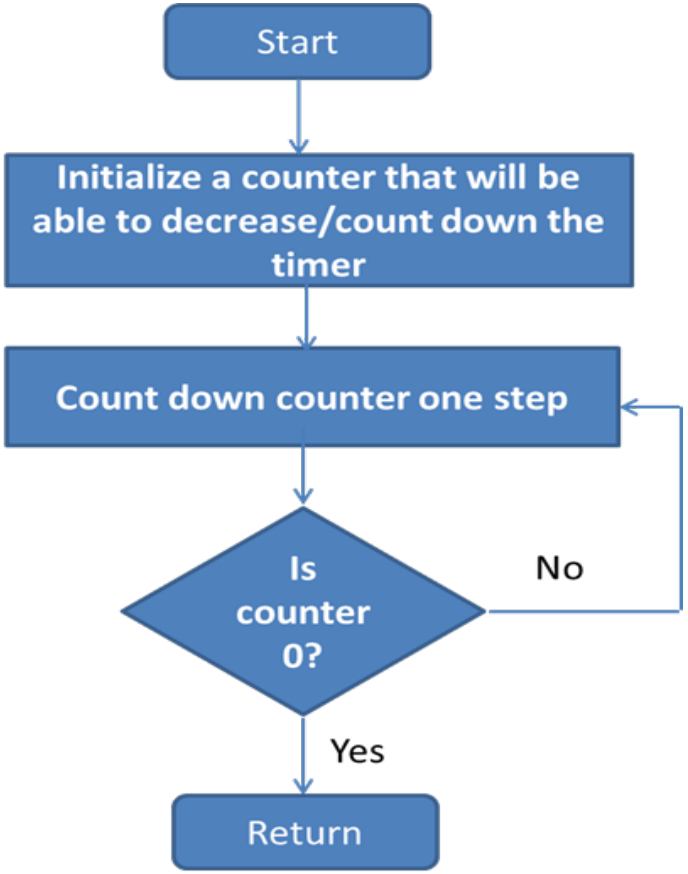
- First lights all the RED LEDs, then turn off the RED LEDs
- Then it ends, but due to some unknown and unidentified reason it goes on n on n on.....

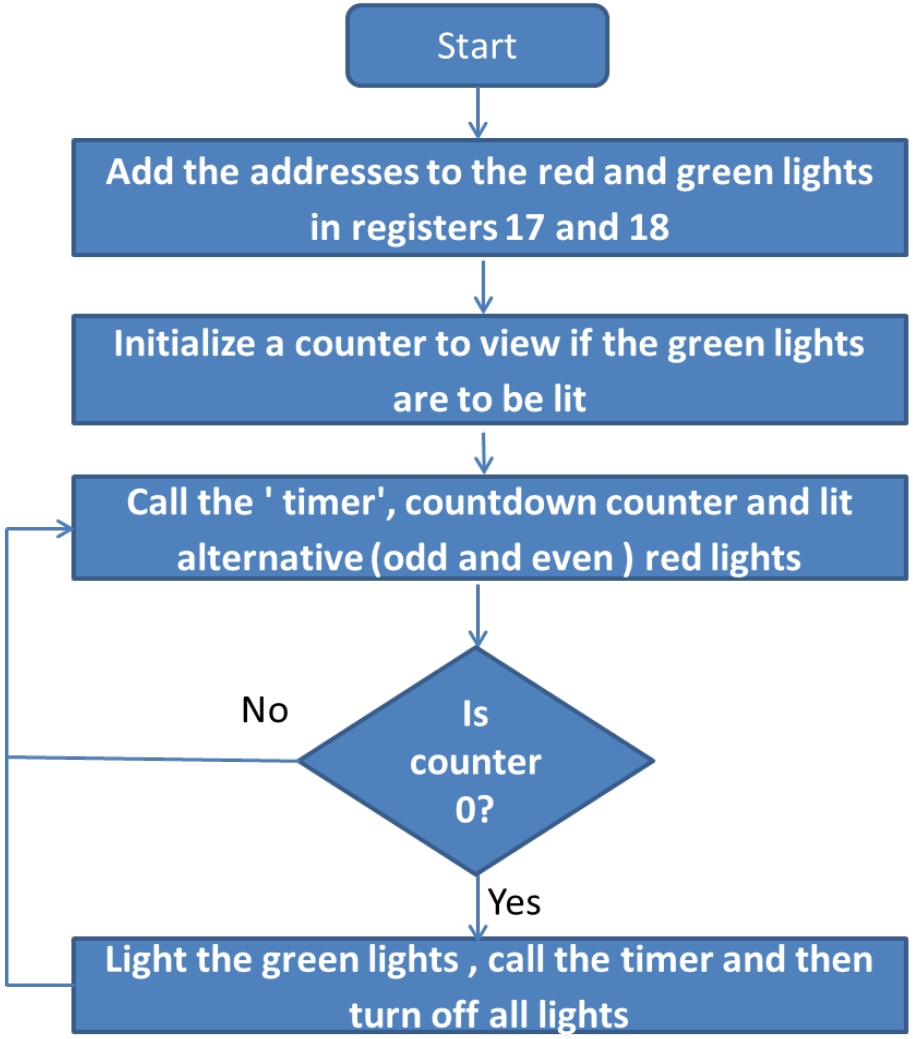
Stockholm University

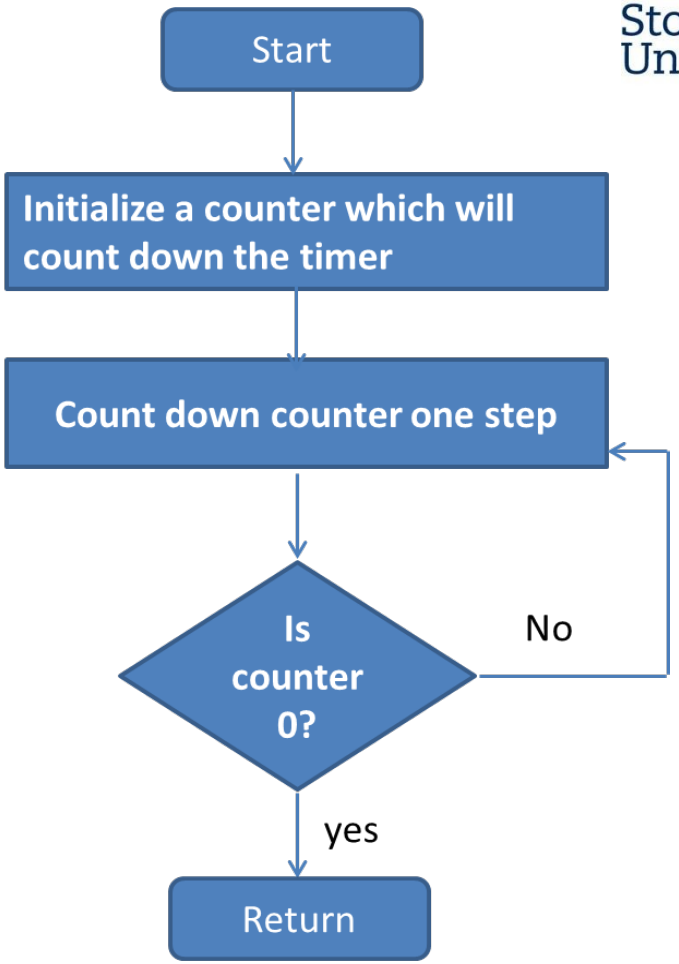
Lab Code

- Check if green LEDs to be lit?
- Turn on and off every other RED LEDs i.e. Odd and even RED LEDs
- Every fifth time turn on the Green LEDs


Start
Add Address to red and green led lights in registers 17 and 18
Add the appropriate value in r8 and lit the red LED lights with its help
Call the subroutine ' timer ' and extinguish the red LED lights using r0
Add the appropriate value in r8 and lit the green led lights with its help
Call the subroutine timer and turn off the green led lights using r0
End


Start
Initialize a counter that will be able to decrease/count down the timer
Count down counter one step
Is counter 0?
No
Yes
Return


Start
Add the addresses to the red and green lights in registers 17 and 18
Initialize a counter to view if the green lights are to be lit
Call the ' timer', countdown counter and lit alternative (odd and even) red lights
Is counter 0?
No
Yes
Light the green lights , call the timer and then turn off all lights


Start
Initialize a counter which will count down the timer
Count down counter one step
Is counter 0?
No
yes
Return

Lab 2

Stockholm University

- Purpose

– Understanding of how the communication of memory mapped buses operate in a computer system

- writing a small program that lets two DE2 cards communicate over serial port