REPORT

ON

"CHOCOLATE VENDING MACHINE"

Prepared in partial fulfillment of the Course

Microprocessor, Programming and Interfacing (CS_EEE_ECE_INSTR F241)

BY

Group 58

Name of the Students:	**ID Nos.:**
Karan Shah	2018B2A70695G
Anant Garg	2018B2A70642G
Ayush Raj	2018B2A70954G
Pungaliya Dhaval Prashant	2018B2A70662G
Jagtar Singh Saggu	2018B2A70911G
Bhavin Soni	2018B2A30685G

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
K K Birla Goa Campus (Semester-II 2020-21)

Table of Contents

User Requirements & Technical Specifications

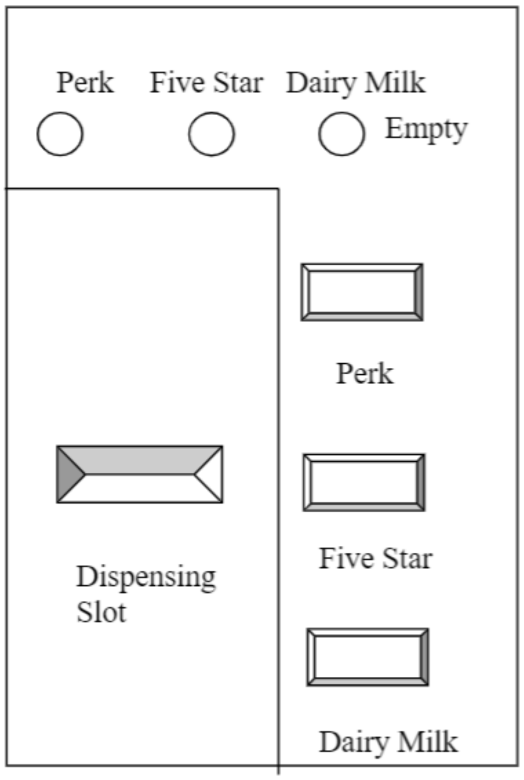
Problem Statement

This automatic machine vends three different types of chocolates.

Perk:	Rs5
Five-Star:	Rs10
Dairy Milk:	Rs20

The currency has to be given in terms of 5 Rupee coins. A weight sensor is used to detect whether the coin is an Rs5 coin or not. There are three buttons available for the selection of the chocolate. After the chocolate has been selected the user has to put the correct currency into the coin slot. When the user has dropped the entire amount into the slot, the machine dispenses the correct chocolate.
LED's are used as indicators to show if any of the chocolates being vended are not available.

User Interface



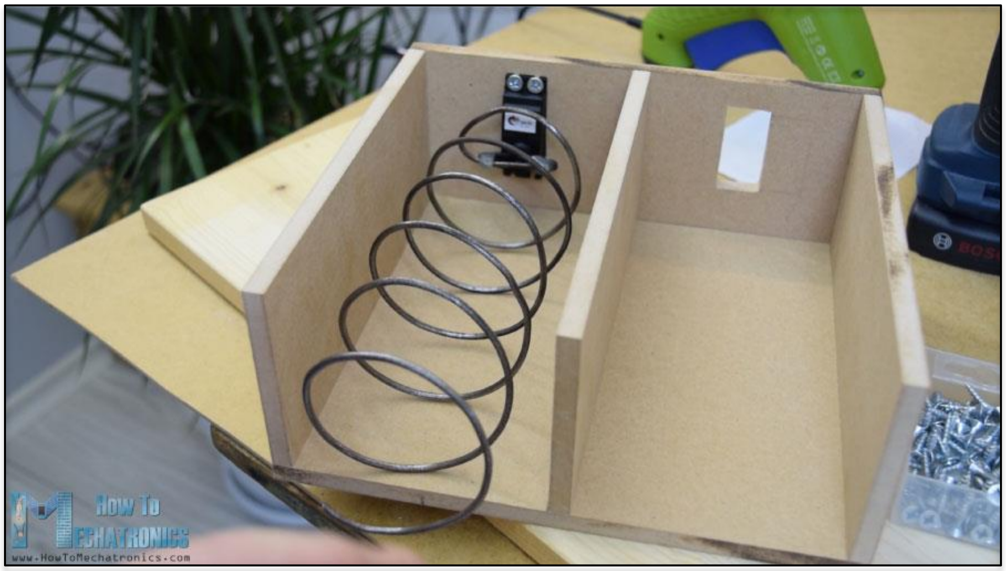
System Description

- There is 1 slot for inserting coins for the chocolate.
- It is assumed that there will be at least around a 0.2s delay between insertions of coins.
- There is a weight sensor that measures the weight of coin and verifies whether the coin is Rs5 coin or not.
- If any other coin is inserted, all coins will be moved to the coin pool and not returned. The system will also flash an LED for around 2secs indicating wrong input of coins.
- If the coins are insufficient, the insufficient coin LED will be ON until the sufficient coins are entered or a wrong input is entered.
- Different chocolates (Perk, Five Star & Dairy Milk) are stored in different places in the machine and are dispensed using different unipolar stepper motors.
- The coins are also moved into the coin bank using a stepper motor.
- Every time a particular chocolate is dispensed its corresponding counter gets decremented by 1.
- LEDs are used to indicate the non-availability of a particular chocolate.

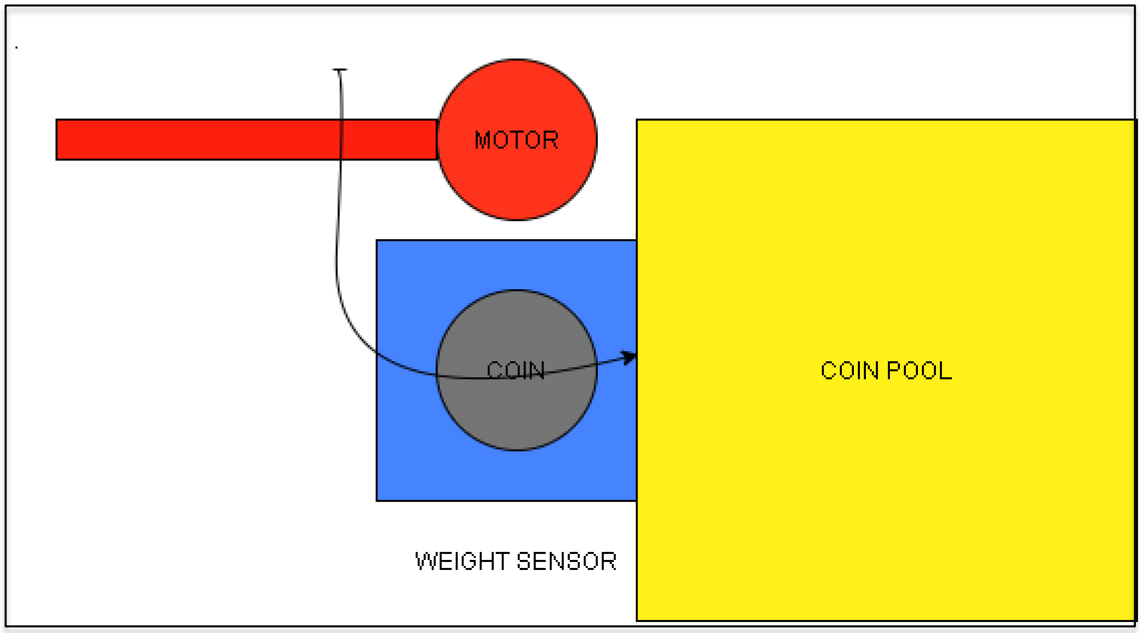
Assumptions & Justifications

Justification

1. Rs5 coin weighs 9gms.
2. Three buttons used for user input (chocolate selection).
3. Due to the low output voltage of the load cell (weight sensor) we have used an amplifier to increase the voltage 500 times.
4. The resolution of an 8-bit ADC is V/256. The range of the load cell is 0-100gms and its resolution is 50mg. Hence we will need an 11-bit ADC. Due to its unavailability we used a 10-bit ADC.
5. We have used 5 LEDs, 3 LEDs to display if the corresponding chocolate is out of stock, 1 LED to display insufficient amount of coins for the particular input which means that the system is expecting more number of 5 rupee coins from the user. The last LED is used to inform the user that a wrong coin has been inputted, this LED will be switched on for around 2 secs.
6. The machine never exits the code unless it is powered off.
7. There are four stepper motors required to be placed, three of the stepper motors are used to dispense the chocolate. We used motor drivers to drive the motor. The motor is connected and rotates one full cycle to dispense the chocolate.



One stepper motor is used to move coins off the weight sensor.



Assumptions

1. The maximum no. of chocolates of a type is 50.
2. No more than 1 button is pressed at a time.
3. If the user deposits the wrong denomination of coins (more or less than required), then the coins are not returned and the user will have to press the button again if required.
4. All Rs5 coins weigh the same.
5. Coins are entered sequentially and there is at least around a 0.2s gap between the 2 inputs.

Working of ADC

The 10-bit ADC that we use has a specific conversion pattern that we must follow. Like other peripheral devices, the write (WR) and read (RD) input signals are valid only when CS is low.

Once CS is low, the onboard system clock permits the conversion to begin with a simple write command and the converted data to be presented to the data bus with a simple read command. The device remains in a sampling (track) mode from the rising edge of EOC until conversion begins with the rising edge of WR, which initiates the hold mode.

After the hold mode begins, the clock controls the conversion automatically. When the conversion is complete, the end-of-conversion (EOC) signal goes low indicating that the digital data has been transferred to the output latch. Lowering CS and RD then resets EOC and transfers the data to the data bus for the processor read cycle.

It is important to note the conversion time is 6µs if no external clk is provided.

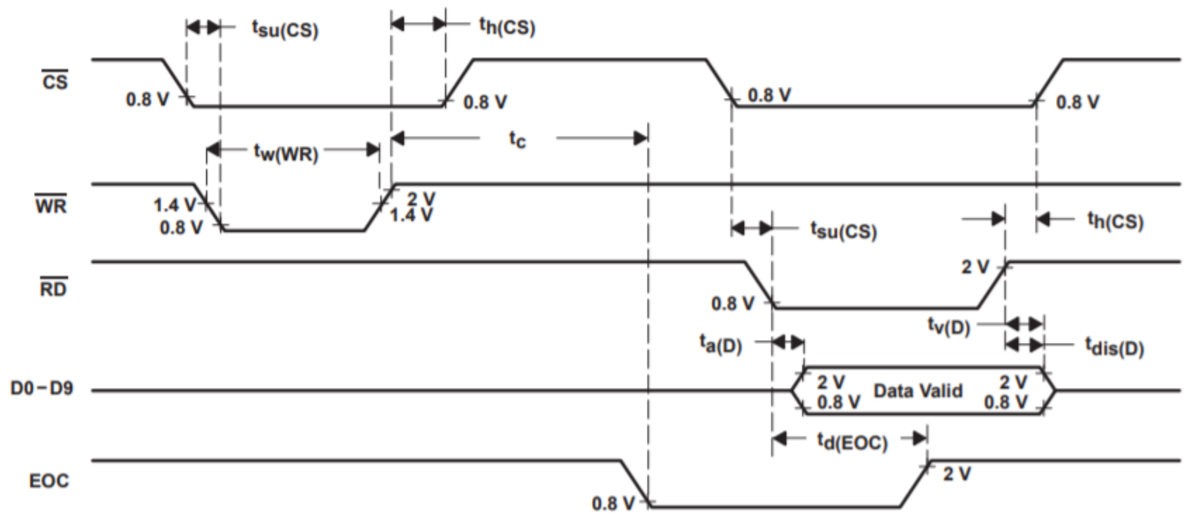


Figure 3. TLC1550 or TLC1551 Operating Sequence

Working of Weight Sensor

The weight sensor we used is a 0-100gm Load Cell and responds to changes in weight of up to 50mg. Force applied onto the weighing pan is converted into electrical signals by the load cell and output to the indicator.

The rated output is 600µV/V and the excitation voltage that we have given is 10V. Hence it will give a maximum output of 6mV. Now this signal is very less so we use an amplifier to amplify it by 500 times, this then goes into the 10-bit ADC and the digital output is sent to 8255.

Components Used
(With justifications wherever Reqd.)

- 8086
- 8284
- 15 MHz Crystal – Given to 8284
- 2716 – Smallest ROM chip available is 2K and as we need to have even and odd bank and ROM is required at reset address which is at FFFF0H and at 00000H where there is the IVT
- 6116 – Smallest RAM chip available is 2K and we need odd and even bank. We need RAM for stack and temporary storage of data
- LS 138, 74155 – Decoder/Demultiplexer
- LS 373, LS 245, LS 244, 74LS244 and required gates.
- 8255 – Interface ADC, motors, LEDs and weight sensor
- RB-Phi-203 – Weight Sensor (Load Cell – Manual Attached)

  Load Range: 0-100gms
  Minimum weight change detected is 50mg
  Voltage Input is 10V
  Four –pin connector

Pin	Signal	Input/ Output	Scale Factor	Description
1	+ Excitation	Input	N/A	Connected to 10V.
2	- Excitation	Input	N/A	Connected to GND
3	+ Signal	Output	3μV/gm	Considering 10V input as rated output varies with input voltage because rated output is 600μV/V. Needs to be scaled up using an amplifier.
4	- Signal	Output	N/A	Output when no load added

- AD624 (Amplifier) – The amplifier gives a 500 times gain because the signal from the Load Cell is very weak. Hence, now the ADC will get a signal with scale factor 15mV/gm.
- TLC1550 (ADC) – It is a 10-bit ADC that converts the amplified signal from the amplifier into digital signal. We have used this ADC because of the resolution that the weight sensor provides – Manual Attached
- 8259 – Interrupts from push buttons will cause the system to jump to the ISR where it will dispense the chocolate after taking the correct amount of coins.
- Push Buttons (SPST) – Used to take selection of chocolate from user.

- LEDs – Notifies the user about lack of a particular chocolate, wrong coin inputted or insufficient coins inputted for the particular chocolate.
- ULN2003A (Motor Driver) – to drive the motors.
- UniPolar Stepper Motor – dispenses chocolates and move the coins off the weight sensor.

Cumulative Table:

NAME	**QUANTITY**
Intel 8086 :Microprocessor (MPU)	1
Intel 8255A : Programmable Peripheral Interface Device (PPI)	2
Intel 8259	1
Intel 8284	1
15 MHz Crystal	1
LS373 (Latch)	3
LS244 (Buffer)	1
LS245 (Bus Transmitter/Receiver)	2
TLC1550 (10-bit ADC)	1
AD624 (Amplifier)	1
RB-Phi-203 (100g micro load cell)	1
OR Gate	18
NOT Gate	8
Push Buttons(SPST)	3
LEDs	5
LS138 Decoder	1
2716 (ROM)	4
6116 (RAM)	2
74155 Decoder/Demultiplexer	2
ULN2003A (Motor Driver)	4
UniPolar Stepper Motor	4
74LS244 (Buffer)	1

Address Map

Memory Map

- 2k - ROM1e (00000-00FFE)

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0

- 2k - ROM1o (00001-00FFF)

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

- 2k - ROM2e (FF000-FFFFE)

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

- 2k - ROM2o (FF001-FFFFF)

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 2k - RAMe (01000-01FFE)

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0

- 2k - RAMo (01001-01FFF)

A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

I/O Map

- ❖ 4B - 8255_A (00h,02h,04h,06h)
- ❖ 4B - 8255_B (08h,0Ah,0Ch,0Eh)
- ❖ 2B - 8259 (80h,82h)

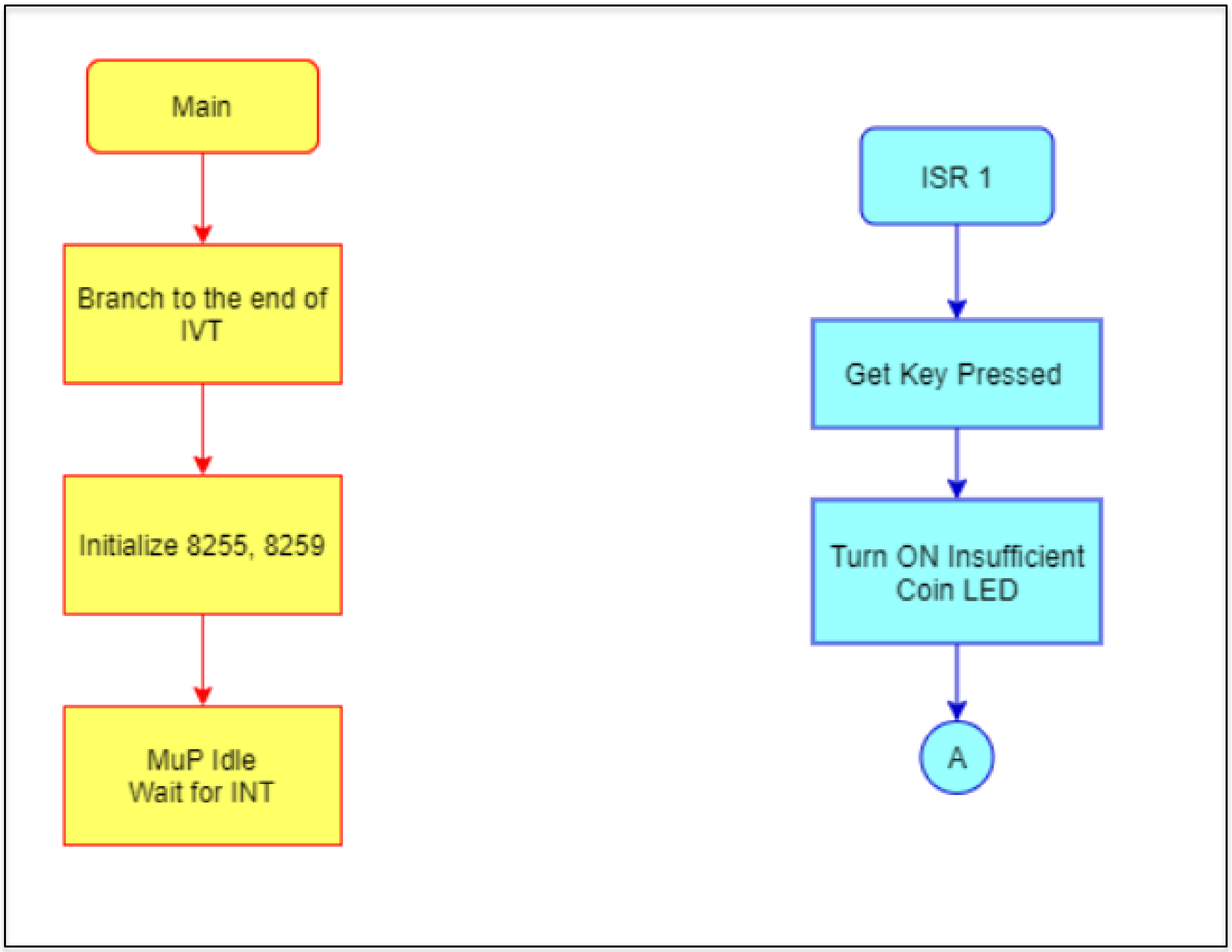
Name of Port/ Register	Input/Output	Address of Port
Port A (8255_A)	Input	00h
Port B (8255_A)	Input	02h
Port C (8255_A)	0-3 Output 4-7 Input	04h
Control Register (8255_A)	N/A	06h
Port A (8255_B)	Output	08h
Port B (8255_B)	Output	0Ah
Port C (8255_B)	Output	0Ch
Control Register (8255_B)	N/A	0Eh
Register 1 (8259)	N/A	80h
Register 2 (8259)	N/A	82h

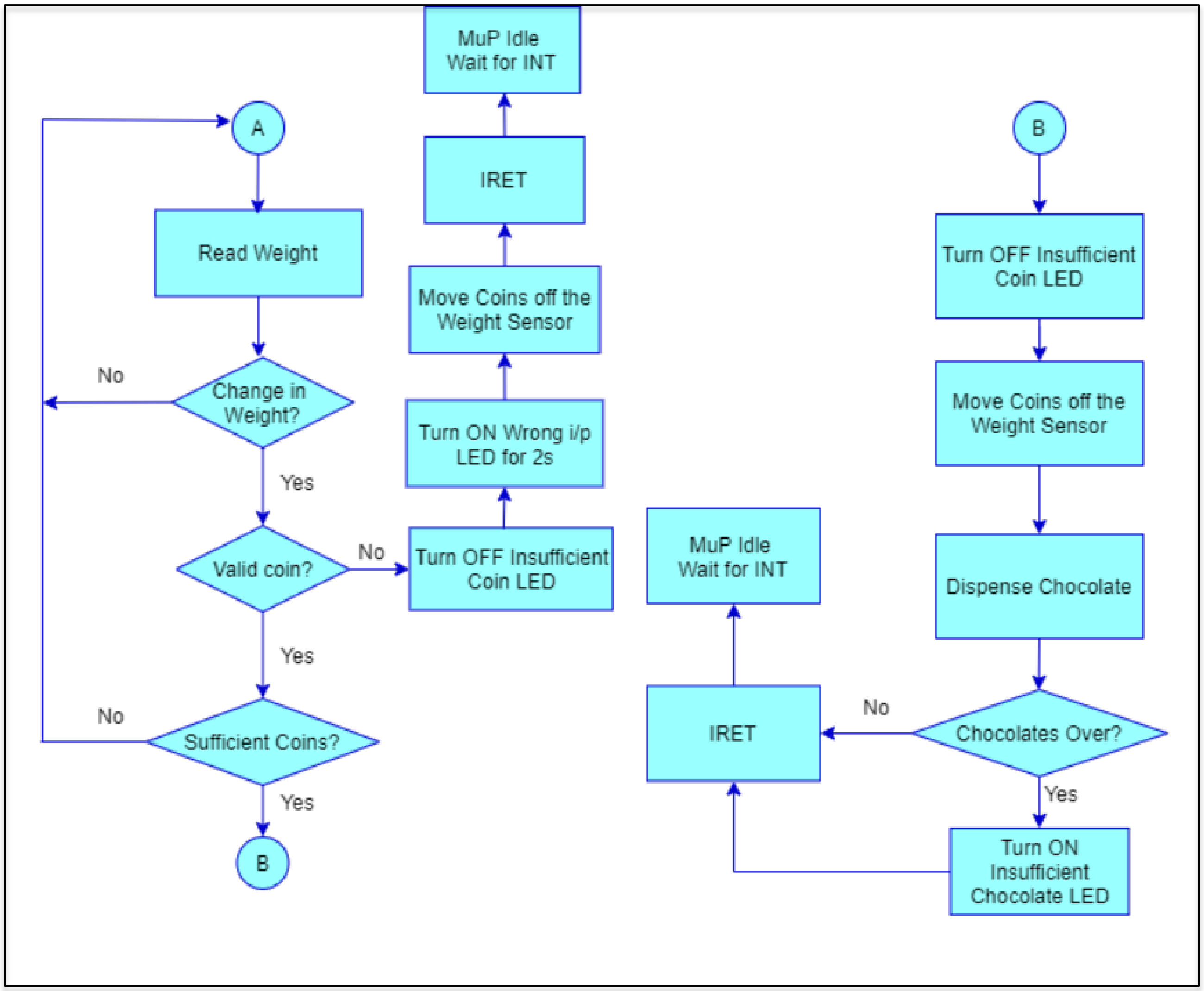
Design

Complete design shown with proper labeling (design attached)

Flowchart

Main Program




MuP Idle Wait for INT
IRET
Move Coins off the Weight Sensor
Turn ON Wrong i/p LED for 2s
Turn OFF Insufficient Coin LED
A
Read Weight
Change in Weight?
No
Yes
Valid coin?
No
Yes
Sufficient Coins?
No
Yes
B
B
Turn OFF Insufficient Coin LED
Move Coins off the Weight Sensor
Dispense Chocolate
Chocolates Over?
No
Yes
Turn ON Insufficient Chocolate LED
IRET
MuP Idle Wait for INT

Variations in Proteus Implementation

1. 2732 instead of 2716 available so changes in code like different start address for RAM and different memory interfacing.
2. As proteus does not have a load cell, we will be using a pressure sensor instead. This pressure sensor does not require an amplifier and has different resolution so the ADC will also change.
3. PortA of ADC is unused because we were taking a 10-bit input in the on-paper design. In the Proteus Design we will be using pressure sensor which is best suited for an 8-bit ADC because of its resolution.
4. An additional Red LED is used instead of a White LED because Proteus only has 4 colors for its normal animated LEDs.
5. 8-bit ADC0808 has a different conversion operations and selection operations due to which number of PORTA outputs of 2nd 8255 will be more in the Proteus Design.
6. Clock is at 2 MHz for 8086.
7. As 8284 is not available so we use normal pulse generation.
8. ROM in only 00000 – as proteus allows to change reset address.
9. As 8259 does not work in proteus NMI is used.

Firmware

Implemented using emu8086, “main.asm” and “main.bin” attached.

List of Attachments

- Complete Hardware Design – “Hardware-Design.pdf”
- Proteus File – “cvm.dsn”
- EMU 8086 ASM files – “main.asm” (on-paper)
- BIN File generated after assembly – “main.bin” (on-paper)
- Manuals:
    - TLC1550 (10-bit ADC)
    - RB-Phi-203 (micro-load cell)
    - AD624 (Amplifier)
    - ULN2003A (Motor Driver)
    - Stepper Motor Working
    - Load-Cell Working