

AVIRUP SAHA

1ST YEAR, M. TECH IT (COURSEWARE ENGINEERING)

ROLL NO – 002030402005

SCHOOL OF EDUCATION TECHNOLOGY

JADAVPUR UNIVERSITY, KOLKATA-700032

**INTEL 8085 & CBT DOCUMENTATION AND MEDIA
MANAGEMENT CATALOGUE**

DATE: 05.08.2021

INTEL 8085 A CBT DOCUMENTATION

8085 A is used for CBT. Used application is Adobe animate CC .1280 X 720 PX dimension stage is used for designing the CBT. The Animate Document (.fla) file size is : 28.1 MB (295,44,026 bytes) and the .exe file size is : 21.8 MB (229,20,919 bytes) .

Location of all the files that are used in CBT is: **C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22.**

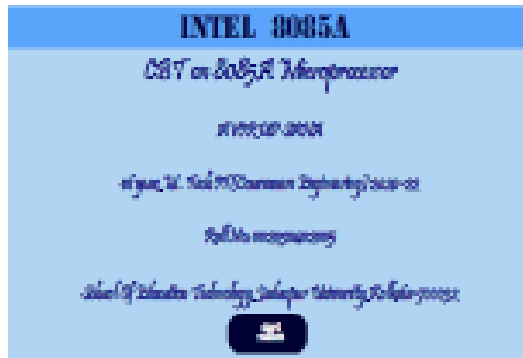
Content with all the details are as below:

Used scene names respectively are:

1. Cover Page
2. Welcome Page
3. Content Page
4. Introduction
5. Pin Diagram
6. ALU
7. Timing and Control unit
8. Registers
9. Data and Address Bus
10. Timing and Control Signal
11. Fetch Operation
12. Execute Operation
13. Machine Cycle
14. Applications
15. Instruction 8085
16. Simulator
17. Coding 1
18. Coding 2
19. Coding 3
20. Quiz 1
21. Quiz 2
22. Quiz 3
23. Matching
24. Sequence
25. End Scene

Cover Page:

Cover page is for my details. This page will provide basic details to differentiate the student name .The layout is as below:



CBT on 8085A Microprocessor

AVIRUP SAHA

1st year, M. Tech IT (Courseware Engineering) 2020-22

Roll No: 002030402005

School Of Education Technology, Jadavpur University,
Kolkata-700032

Welcome Page

This page will show “Welcome to Intel 8085 tutorial!” There is sub section with Objective, Target Audience, Hardware and Software requirements, User Guide and About Us

➤ Objective

CBT or Computer based tutorials contain training module that help us to import training in interactive and entertaining manner .Modules are designed in a way to make learning more convenient and effortless .In this CBT the total learning will be more interactive and effective to the target audience .Here Intel8085A will be covered.



➤ Target Audience

This computer based tutorial is designed for the Under Graduate Electronics and Electrical students. Besides that students who are interested in Microprocessor and Microcontroller specially want to know about the 8085A architecture and its application based usage.

➤ Hardware and Software requirements

The software and the hardware these are used for designing the CBT is as below

REQUIRED SOFTWARE

Adobe animate CC, Microsoft power point, Sound forge etc.

REQUIRED HARDWARE

Processor: Intel Pentium 4 or Intel Centrino, Intel Xeon, or Intel Core Duo (or compatible) processor (2GHz or faster processor)

Operating system: Windows 10 version V1903, V1909, V2004 and later.

RAM: 2GB RAM (8GB recommended)

Hard disk space: 4GB of available hard-disk space for installation; more free space required during installation (cannot install on removable flash storage devices)

Monitor resolution: 1024x900 display (1280x1024 recommended)

GPU: OpenGL version 3.3 or higher (DirectX 12 with feature level 12_0 recommended)

➤ **User guide**

The user guide part will help the user to navigate through the whole CBT. For that user will be given instructions accordingly so that he/she will not get confused while attending the tutorial. The content of the user guide is as follows:

Please read the instructions carefully

- 1> Press the sound button immediately once navigated to a certain page.
- 2> Sound button to be used manually to get the audio of that page.
- 3> Once you are in content page you can't come to this welcome page.
- 4> You can access coding, simulator and quiz once tutorial is finished.
- 5> While using simulator please read the instructions carefully and press the button accordingly to execute the operation.
- 6> Simulator, coding and quiz. Once you are in any of these sections you will be no longer able to return tutorial page.

All the best for the tutorial!

➤ **About us**

We are the students of Master of Technology with specialization in Information technology (Courseware engineering), Jadavpur University

The CBT is created on 26.07.2021

Content Page

The content page will navigate the user to all tabs. There are responsive buttons for that. Home button will take the user to the Content page. Back will navigate to next section and previous button will take to previous page. Once user is in content page he/she will not be able to go to welcome Page or Cover Page. Sound button is there on the left bottom side so enable and disabling audio

Audio will guide the user to go through the section. Coding, Simulator and Quiz is not available until the tutorial is finished



Introduction

The introduction page will describe the details of the Intel 8085 A. An audio file is also used for that. Masking is used for the fade in effect in Adobe animate. The content is as follows:

Intel 8085 is an 8-bit, NMOS microprocessor.

It is a 40 pin C package fabricated on a single LSI chip.

The Intel 8085A uses a single +5V D.C supply for its operation.

Its clock speed is about 3 MHz .The clock cycle is of 320 ns. The time for the back cycle of the Intel 8085 A-2 is 200 ns.

It has 80 basic instructions and 246 opcodes.

Intel 8085A consists of three main sections, an arithmetic and logic unit a timing and control unit and several registers

Pin Diagram

Pin Diagram part is a movie clip where the details of the pins are shared in details with diagram as requirement. The pin diagram is designed in Microsoft Power point .Sound is used for navigating the diagram.

ALU

The arithmetic and logic unit, ALU performs arithmetic and logic operations. The content is as follows:

1. Addition 2. Subtraction 3. Logical AND 4. Logical OR 5. Logical EXCLUSIVE OR 6. Complement (logical NOT) 7. Increment (add 1) 8. Decrement (subtract 1) 9. Left shift (add input to itself) 10. Clear (result is zero)

Timing and Control unit

The timing control unit section is used with the timing and control unit .The content is as follows:
The timing and control unit is a section of the CPU. It generates timing and control signals which are necessary for the execution of instructions. It controls provides status, control and timing signals which are required for the operation of memory and I/O 2 devices. It controls the entire operation of the microprocessor and peripherals consented to it. Thus it is seen that control unit of the CPU acts as a brain of the computer.

Registers

After timing control unit and register section is there .The content is as follows:

Registers are small memories within the CPU. They are used by the microprocessor for temporary storage and manipulation of data and instructions. Data remain in the registers till they are sent to the memory or I/O devices. Intel 8085 microprocessor has the following registers.

1. One 8-bit accumulator (ACC) i.e. register A.
2. Six 8-bit general purpose registers. These are B, C, D, E, H and L.

3. One 16-bit program counter, PC.
4. Instruction register
5. Status register
6. Temporary register.

Status flags also there in Intel 8085. These are:

1. CARRY (CS)
2. ZERO (Z)
3. SIGN (S)
4. PARITY (P)
5. AUXILIARY CARRY (AC)

Data and Address Bus

Data and address bus section will show the multiplexing in 8085A. The content is as follows:

A8-A15 Higher Order Address bus: These are o/p tri-state (a state of high impedance) signals used as higher order 8 bits of 16 bit address. These signals are unidirectional and are given from 8085 to select memory or I/O devices.

AD0-AD7 Multiplexed Address/Data bus: These are I/O tri-state signals, having 2 sets of signals. They are address and data. The lower 8 bit of 16 bit address is multiplexed/time shared with data bus.

Timing and Control Signal

The timing and control signal shows the mechanism for generating timing signals for the execution of instruction and control of peripheral devices. The content is as follows:

For the execution of an instruction a μP fetches the instruction from the memory and executes it. The time taken for the execution of an instruction is called instruction cycle (IC).

An instruction cycle consists of a fetch cycle (FC) and an execute cycle (EC). A fetch cycle is the time required for the fetch operation in which the machine code of the instruction (opcode) is fetched from the memory.

An execute cycle is of variable width which depends on the instruction to be executed. The total time for the execution is given by $IC = FC + EC$

Fetch Operation

The fetch operation of the 8085A is shown here. It is as follows. In fetch operation the μP gets the 1st byte of the instruction, which is operation code (opcode), from the memory.

The total time for fetch operation is the time required for fetching an opcode from the memory. This time is called fetch cycle.

If the memory is slow, it may take more time. In that case the μP has to wait for some time till it receives the opcode from the memory. The time for which the μP waits is called wait cycle.

Execute Operation

Execution Operation describes the execution operation of the 8085A. The content as follows:

The opcode fetched from the memory goes to the data register, DR (data/address buffer in Intel 8085) and then to instruction register, IR. From the instruction register it goes to the decoder circuitry which is within the microprocessor. After the instruction is decoded, execution begins. If the operand is in the general purpose registers, execution is immediately performed. The time taken in decoding and the address of the data, some

read cycles are also necessary to receive the data from the memory. These read cycle are similar to opcode fetch cycle. The fetch quantities in these cycles are address or data.

Machine Cycle

The machine cycle describes the instruction cycle. An instruction cycle consists of one or more machine cycles as shown in diagram. The audio button also guides the user regarding machine cycle. Content also shared .MVI instruction is shared here. A machine cycle consists of a number of clock cycles. One clock cycle is known as state

Applications

Applications of the microprocessor are describes here.The content is as follows:

How Microprocessor are being used for numerous applications

1. Personal Computer
2. Numerical Control
3. Mobile Phones
4. Automobiles
5. Bending Machines
6. Medical Diagnostic Equipment
7. Automatic voice recognizing systems
8. Prosthetics
9. Traffic light Control
10. Entertainment Games
11. Digital Signal Processing
12. Communication terminals
13. Process Control
14. Calculators
15. Sophisticated Instruments
16. Telecommunication Switching Systems
17. Automatic Test Systems.

Instruction 8085

8085 instruction sections help the user to know all the variation of the instructions that are used for coding in Intel 8085 .Here five tabs created as button for instruction part.

The content is as follows:

Data Transfer Group

1. MOV r1, r2 (Move Data; Move the content of the one register to another): [r1] <-- [r2]
2. MOV r, m (Move the content of memory register): r <-- [M]
3. MOV M, r. (Move the content of register to memory): M <-- [r]
4. MVI r, data. (Move immediate data to register): [r] <-- data
5. MVI M, data. (Move immediate data to memory): M <-- data
6. LXI rp, data 16. (Load register pair immediate): [rp] <-- data 16 bits, [rh]<-- 8 LSBs of data
7. LDA addr. (Load Accumulator direct): [A] <-- [addr]
8. STA addr. (Store accumulator direct): [addr] <-- [A]
9. LHLD addr. (Load H-L pair direct): [L] <-- [addr], [H] <-- [addr+1]



10. SHLD addr. (Store H-L pair direct): [addr] <-- [L],
[addr+1] <-- [H]
11. LDAX rp. (LOAD accumulator indirect): [A] <-- [[rp]]
12. STAX rp. (Store accumulator indirect) : [[rp]] <-- [A]
13. XCHG. (Exchange the contents of H-L with D-E pair) :
[H-L] <--> [D-E]

Arithmetic Group

1. ADD r. (Add register to accumulator): $[A] \leftarrow [A] + [r]$
2. ADD M. (Add memory to accumulator): $[A] \leftarrow [A] + [[H-L]]$
3. ADC r. (Add register with carry to accumulator): $[A] \leftarrow [A] + [r] + [CS]$
4. ADC M. (Add memory with carry to accumulator): $[A] \leftarrow [A] + [[H-L]][CS]$
5. ADI data (Add immediate data to accumulator): $[A] \leftarrow [A] + \text{data}$
6. ACI data (Add with carry immediate data to accumulator): $[A] \leftarrow [A] + \text{data} + [CS]$
7. DAD rp. (Add register pair to H-L pair): $[H-L] \leftarrow [H-L] + [rp]$
8. SUB r. (Subtract register from accumulator): $[A] \leftarrow [A] - [r]$
9. SUB M. (Subtract memory from accumulator): $[A] \leftarrow [A] - [[H-L]]$
10. SBB r. (Subtract register from accumulator with borrow): $[A] \leftarrow [A] - [r] - [CS]$
11. SBB M. (Subtract memory from accumulator with borrow): $[A] \leftarrow [A] - [[H-L]] - [CS]$
12. SUI data. (Subtract immediate data from accumulator): $[A] \leftarrow [A] - \text{data}$
13. SBI data. (Subtract immediate data from accumulator with borrow): $[A] \leftarrow [A] - \text{data} - [CS]$
14. INR r (Increment register content): $[r] \leftarrow [r] + 1$
15. INR M. (Increment memory content): $[[H-L]] \leftarrow [[H-L]] + 1$
16. DCR r. (Decrement register content). $[r] \leftarrow [r] - 1$
17. DCR M. (Decrement memory content): $[[H-L]] \leftarrow [[H-L]] - 1$
18. INX rp. (Increment register pair) $[rp] \leftarrow [rp] + 1$
19. DCX rp (Decrement register pair) : $[rp] \leftarrow [rp] - 1$
20. DAA (Decimal adjust accumulator) .

Logical Group

1. ANA r. (AND register with accumulator) : $[A] \leftarrow [A] \wedge [r]$
 2. ANA M. (AND memory with accumulator) : $[A] \leftarrow [A] \wedge [[H-L]]$
 3. ANI data. (AND immediate data with accumulator): $[A] \leftarrow [A] \wedge \text{data}$
 4. ORA r. (OR register with accumulator) : $[A] \leftarrow [A] \vee [r]$
 5. ORA M. (OR memory with accumulator): $[A] \leftarrow [A] \vee [[H-L]]$
 6. ORI data. (OR immediate data with accumulator) : $[A] \leftarrow [A] \vee \text{data}$
 7. XRA r. (EXCLUSIVE – OR register with accumulator) : $[A] \leftarrow [A] \vee [r]$
 8. XRA M. (EXCLUSIVE-OR memory with accumulator) : $[A] \leftarrow [A] \vee [[H-L]]$
 9. XRI data. (EXCLUSIVE-OR immediate data with accumulator): $[A] \leftarrow [A] \vee \text{data}$
 10. CMA. (Complement the accumulator) : $[A] \leftarrow \neg [A]$
 11. CMC. (Complement the carry status): $[CS] \leftarrow \neg [CS]$
 12. STC. (Set carry status): $[CS] \leftarrow 1$
 13. CMP r. (Compare register with accumulator) : $[A] - [r]$
 14. CMP M. (Compare memory with accumulator) : $[A] - [[H-L]]$
 15. CPI data. (Compare immediate data with accumulator) : $[A] - \text{data}$
 16. RLC (Rotate accumulator left): $[An+1] \leftarrow [An], [A0] \leftarrow [A7], [CS] \leftarrow [A7]$
- The content of the accumulator is rotated left by one bit. The seventh bit

of the accumulator is moved to carry bit as well as to the zero bit of the accumulator. Only CS flag is affected.

17. RRC (Rotate accumulator right) : $[A7] \leftarrow [A0]$, $[CS] \leftarrow [A0]$, $[An] \leftarrow [An+1]$

The content of the accumulator is rotated right by one bit. The zero bit of the accumulator is moved to the seventh bit as well as to carry bit. Only CS flag is affected.

18. RAL (Rotate accumulator left through carry) : $[An+1] \leftarrow [An]$, $[CS] \leftarrow [A7]$, $[A0] \leftarrow [CS]$

19. RAR (Rotate accumulator right through carry) : $[An] \leftarrow [An+1]$, $[CS] \leftarrow [A0]$, $[A7] \leftarrow [CS]$

Branch Group

1. JMP addr (label) (Unconditional jump: jump to the instruction specified by the address):
 $[PC] \leftarrow \text{Label}$

2. Conditional Jump addr (label):

After the execution of the conditional jump instruction the program jumps to the instruction specified by the address (label) if the specified condition is fulfilled.

The program proceeds further in the normal sequence if the specified condition is not fulfilled.

If the condition is true and program jumps to the specified label, the execution of a conditional jump takes 3 machine cycles: 10 states.

If condition is not true, only 2 machine cycles; 7 states are required for the execution of the instruction.

Conditional Jump addr (label) are:

JZ addr (label). (Jump if the result is zero)

JNZ addr (label) (Jump if the result is not zero)

JC addr (label). (Jump if there is a carry)

JNC addr (label). (Jump if there is no carry)

JP addr (label). (Jump if the result is plus)

JM addr (label). (Jump if the result is minus)

JPE addr (label) (Jump if even parity)

JPO addr (label) (Jump if odd parity)

3. CALL addr (label) (Unconditional CALL: call the subroutine identified by the operand)

CALL instruction is used to call a subroutine.

Before the control is transferred to the subroutine, the address of the next instruction of the main program is saved in the stack.

The content of the stack pointer is decremented by two to indicate the new stack top.

Then the program jumps to subroutine starting at address specified by the label.

4. RET (Return from subroutine)

5. RST n (Restart) Restart is a one-word CALL instruction.

The content of the program counter is saved in the stack.

The program jumps to the instruction starting at restart location.

Stack, I/O and Machine Control Group

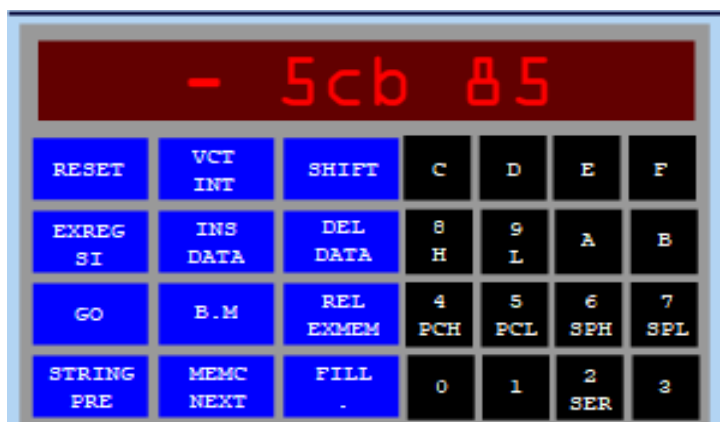
1. IN port-address. (Input to accumulator from I/O port) [A] <-- [Port]
2. OUT port-address (Output from accumulator to I/O port) [Port] <-- [A]
3. PUSH rp (Push the content of register pair to stack)
4. PUSH PSW (PUSH Processor Status Word)
5. POP rp (Pop the content of register pair, which was saved, from the stack)
6. POP PSW (Pop Processor Status Word)
7. HLT (Halt)

Stack, I/O and Machine Control Group

8. XTHL (Exchange stack-top with H-L)
9. SPHL (Move the contents of H-L pair to stack pointer)
10. EI (Enable Interrupts)
11. DI (Disable Interrupts)
12. SIM (Set Interrupt Masks)
13. RIM (Read Interrupt Masks)
14. NOP (No Operation)

Simulator

The simulator part is associated in this CBT for the user interactivity. The user will get to know how the 8085 A kit works .How to operations are done using opcodes and coding instruction set. The user will come to know real world usage of Intel 8085A microprocessor



The sound part will guide the user for operation .The user have to press the button as per instructions to understand the operation .Here addition is shown as example.

Fig: Simulator interface (Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

This image is taken from simulator layer of adobe animate CC .The location is: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file

Here interface of the simulator is also shown

Coding 1

User will code first program. After submission if wrong it will show the correct answer to the user

Coding 2

Same as coding 1

Coding 3

Same as coding 1

Coding1

Write a program for 8085A to add two 16 bit numbers using 8 bit operation

Ans:

```
LHLD 4000H
XCHG
LHLD 4002H
MOV A, E
ADD L
MOV L, A
MOV A, D
ADC H
MOV H, A
SHLD 4004H
HLT
```

To be written as:

LHLD 4000H/XCHG/LHLD 4002H/MOV A, E/ADD L/MOV L, A/MOV A, D/ADC H/MOV H, A/SHLD 4004H/HLT

Coding 2

Write a program to subtract two 8-bit numbers

Ans:

```
LXI H 4000H
MOV A, M
INX H
SUB M
INX H
MOV M, A
HLT
```

To be written as:

LXI H 4000H/MOV A, M/INX H /SUB M/INX H/MOV M, A/HLT

Coding 3

Write a program to exchange the contents of memory locations 2000H and 4000H using indirect addressing instructions.

Ans:

```
LXI H 2000H
LXI D 4000H
MOV B, M
LDAX D
MOV M, A
MOV A, B
STAX D
HLT
```

To be written as:

LXI H 2000H /LXI D 4000H/MOV B, M /LDAX D /MOV M, A /MOV A, B /STAX D/HLT

Quiz 1

This will be a mcq part. Here the user have to answer MCQ questions. One attempt available and after that user will not able to get back previous one and correct answer will be shared once answer is given

Quiz 2

Same as quiz 1

Quiz 3

Same as quiz 1

Priority interrupts of 8085 is :

- ☐ A. TRAP > INTR > RST 7.5 > RST 6.5 > RST 5.5
- ☐ B. RST 7.5 > RST 6.5 > RST 5.5 > TRAP > INTR
- ☐ C. TRAP > RST 7.5 > RST 6.5 > RST 5.5 > INTR
- ☐ D. RST 7.5 > RST 6.5 > RST 5.5 > INTR > TRAP

Q1. Ans: A

Example of Arithmetic group is :

- ☐ A. MOV , MVI , LXI
- ☐ B. ADD , SUB , INR.
- ☐ C. ANA , XRA , CMP
- ☐ D. JMP , JNZ , CALL

Q2. Ans: B

How many operations are there in the instruction set of 8085 microprocessor?

- ☐ A. There are 74 operations in the 8085 microprocessor.
- ☐ B. There are 73 operations in the 8085 microprocessor.
- ☐ C. There are 76 operations in the 8085 microprocessor.
- ☐ D. There are 72 operations in the 8085 microprocessor.

Q3 Ans: A

Fig: Quiz 1 page design

(Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

Fig: Quiz 2 page design

(Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

Fig: Quiz 3 page design

(Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

Matching



This is a drag and drop operated quiz where user have to drag the option from left side to correct match of right side .The user have to match all options correct then will get answer as correct .User can do reset if he did one wrong and want to try again. User can use the sound to understand the instruction of the process as well

Fig: Match the following page design

(Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

Sequence



In sequence portion user have to answer correct opt codes or operation code that are used while coding. From drop down he/she have to do that .All the correct answer will be show at once after clicking the button that is instructed to press for .There is an end session button also to finish the tutorial

Fig: Choose the current sequence

(Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

End Scene



The end scene congratulates the user for completing the session of CBT successfully .A back ground sound also there

Fig: End Scene

(Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file)

Legend

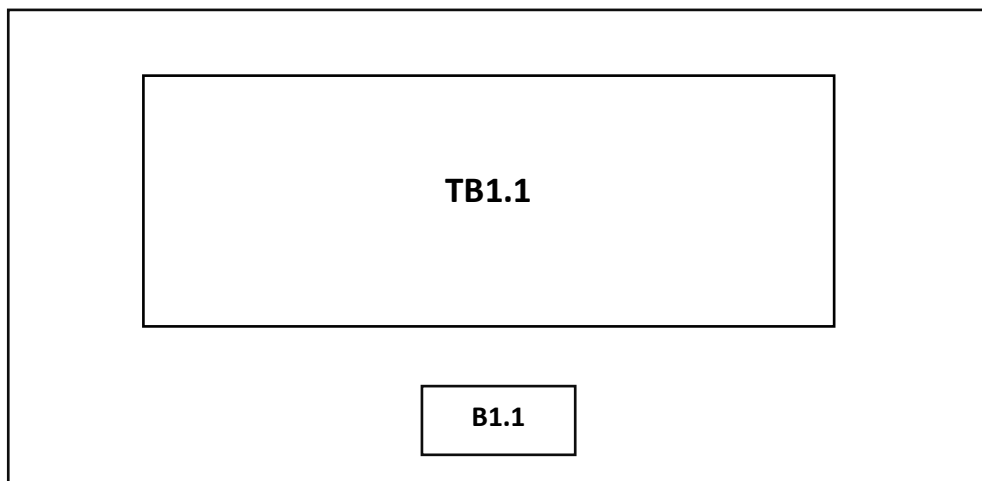
Sc: Scene
IA: Interactive Animation
B: Button
BM: Background Music
BI: Background Image
TB: Text Box
AT: Animated Text

Scene design page that is shared below in left image (Location: **C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\.fla_file**) and story boarding

/*Sc1:*/

TB1.1: Cover page content
B1.1: Start button

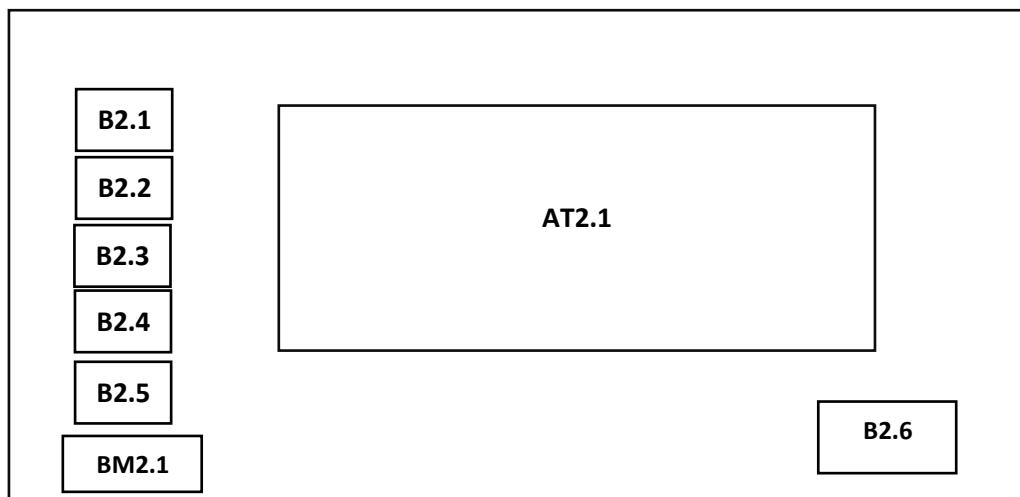
This page is designed as the cover page of the CBT. In TB1.1 the details of the student is there and B1.1 is to start the CBT



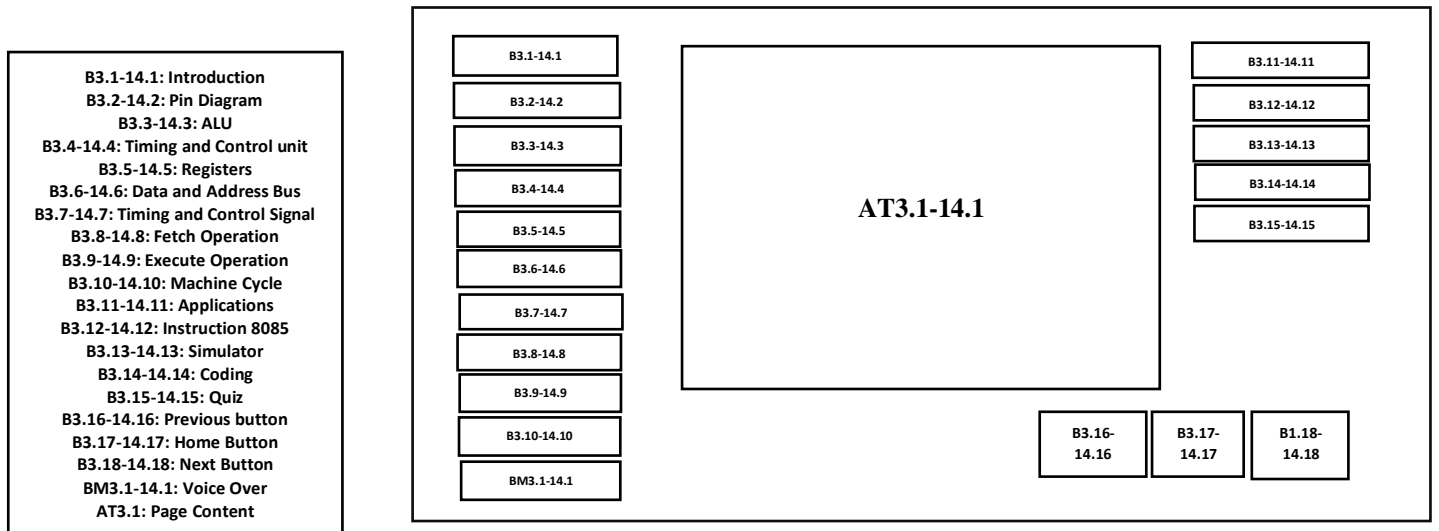
/*Sc2:*/

B2.1-14.1: Introduction
B2.2-14.2: Pin Diagram
B2.6: Next Button
B2.3-14.3: ALU
BM2.1: Next Button
AT2.1: Page Content

This page is as welcome page where B2.1 is for the Objective of the CBT. Then pressing B2.2 will show Target Audience in AT2.1 the B2.3 is for the Hardware and software requirements. B2.4 is for User guide and B2.5 is for the description of About Us. B2.6 is for the next button and BM2.1 is for the Voice over for the instructions to user regarding the CBT. User have to press the BM2.1 at the beginning of this Scene 2(Sc2)



/*Sc3 to Sc14 (all same):*/



/***** Please go to next page for the brief details of the story boarding *****/

B3.1-14.1: Introduction

These buttons are to navigate the user to the introduction page from any of the scene of tutorial portion .The user will see the content of the page on the AT3.1 portion.

B3.2-14.2: Pin Diagram

These buttons are to help to user to go to Pin Diagram page from any of the scene of tutorial portion. In AT4.1 user will see the content of the Pin Diagram page. The Slide is used for the description of all the 40 pins at 8fps

B3.3-14.3: ALU

For the Arithmetic and logic unit portion of the CBT B3.3-14.3 is used .In AT5.1 the user will be shared the details of the ALU. User can navigate to the ALU section from any of the tutorial part

B3.4-14.4: Timing and Control unit

From B3.4 to 14.4 is for the Timing and Control unit section navigation. The content of the button will be shown in AT6.1

B3.5-14.5: Registers

Using these button user can proceed for the brief description of the register part .These buttons are available up to tutorial portion and content will be shown in AT7.1

B3.6-14.6: Data and Address Bus

These buttons are to navigate the user to the Data and Address Bus page from any of the scene of tutorial portion .The user will see the content of the page on the AT8.1 portion.

B3.7-14.7: Timing and Control Signal

For the Timing and Control Signal portion of the CBT B3.7-14.7 is used .In AT9.1 the user will be shared the details of the Timing and Control Signal. User can navigate to the Timing and Control Signal section from any of the tutorial part

B3.8-14.8: Fetch Operation

For the Fetch Operation portion of the CBT B3.8-14.8 is used .In AT10.1 the user will be shared the details of the Fetch Operation and user can use these buttons in any of the tutorial part

B3.9-14.9: Execute Operation

Using these button user can proceed for the brief description of the Execute Operation part .These buttons are available up to tutorial portion and content will be shown in AT11.1

B3.10-14.10: Machine Cycle

For the Machine Cycle portion of the CBT B3.10-14.10 is used .In AT12.1 the user will be shared the details of the Machine Cycle. User can navigate to the Machine Cycle section from any of the tutorial part

B3.11-14.11: Applications

These buttons will lead to the Applications portion of the CBT. Using B3.11-14.11 in any of the tutorial part at AT13.1 the user will see the detailed Applications of 8085 A.

B3.12-14.12: Instruction 8085

Instruction 8085 button will take the user to the detailed Instruction 8085 portion.AT14.1 the brief description will be there .Sc15 has other navigation button inAT14.1 as well .User can go to any of the tutorial portion and once user reach this section after that user will get access of the Simulator ,Coding and Quiz section. Except this section user will not be allowed to go to the Simulator (Sc16), Coding (Sc17-Sc19) and Quiz (Sc20-24) section.

B3.13-14.13: Simulator

These buttons will have popup but no navigation is available. From 3.13-14.3 the user cannot go to this section .Once the user visits Scene15 then can go to any of the Simulator (Sc16), Coding (Sc17-Sc19) and Quiz (Sc20-24) section.

B3.14-14.14: Coding

These buttons will have popup but no navigation is available. From 3.14-14.14 the user cannot go to this section .Once the user visits Scene15 then can go to any of the Simulator (Sc16), Coding (Sc17-Sc19) and Quiz (Sc20-24) section.

B3.15-14.15: Quiz

These buttons will have popup but no navigation is available. From 3.15-14.15 the user cannot go to this section .Once the user visits Scene15 then can go to any of the Simulator (Sc16), Coding (Sc17-Sc19) and Quiz (Sc20-24) section.

B3.16-14.16: Previous button

These buttons will navigate the user to the previous scene except Sc.3. For Sc3 previous button it will lead to itself rest will lead to previous scene. For example: Sc.4 previous button will lead to Sc.3, Sc.5 previous button will lead to Sc.4, Sc.6 previous button will lead to Sc.5 and so on

B3.17-14.17: Home Button

These buttons will navigate the user to the Content page scene (Sc. 3). Only Sc.3 will lead to itself rest will lead to content page scene (Sc.3) .For example: Sc.4 will lead to Sc.3, Sc.5will lead to Sc.3 and so on

B3.18-14.18: Next Button

These buttons will navigate the user to the next scene. For example: Sc.3 next button will lead to Sc.4, Sc.4 next button will lead to Sc.5, Sc.5 next button will lead to Sc.6 and so on

BM3.1-14.1: Voice Over

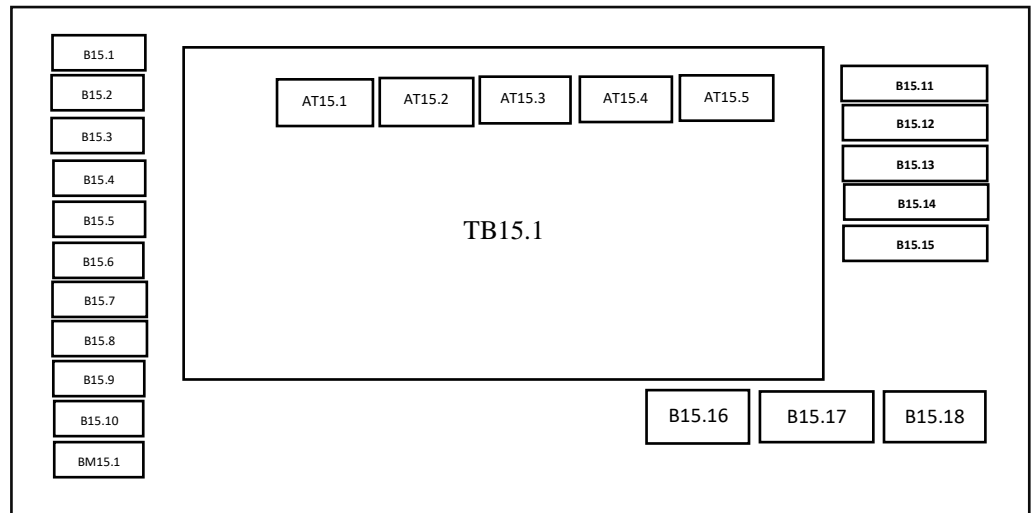
These BM3.1-14.1 buttons will trigger the audio file that is embedded to that scene .These button are start and stop button. User have to press the button manually at the beginning of the section. User have to start and stop the button manually. These voice over will guide the user regarding the section. It will navigate the user and will share the content of the scene as well.

AT3.1-14.1: Page Content

These buttons are for the description of the section .Using button the user can switch the tabs and the button that is pressed accordingly the content will be shown at AT 3.1 to 14.1.These are animate text that are used using masking to give fade-in effect in text

/*Sc15:*/

B15.1: Introduction
B15.2: Pin Diagram
B15.3: ALU
B15.4: Timing and Control unit
B15.5: Registers
B15.6: Data and Address Bus
B15.7: Timing and Control Signal
B15.8: Fetch Operation
B15.9: Execute Operation
B15.10: Machine Cycle
B15.11: Applications
B15.12: Instruction 8085
B15.13: Simulator
B15.14: Coding
B15.15: Quiz
B3.16-14.16: Previous button
B3.17-14.17: Home Button
B3.18-14.18: Next Button
BM3.1-14.1: Voice Over
TB15.1: Page Content
AT15.1: Data Transfer Group
AT15.2: Arithmetic Group
AT15.3: Logical Group
AT15.4: Branch Group
AT15.5: Stack, I/O and Machine Control Group



B15.1: Introduction

This button is to navigate the user to the introduction page. The user will see the content of the page on the AT3.1 portion.

B15.2: Pin Diagram

This button is to help user to go to Pin Diagram page .In AT4.1 user will see the content of the Pin Diagram page. The Slide is used for the description of all the 40 pins at 8fps

B15.3: ALU

For the Arithmetic and logic unit portion of the CBT 15.3 is used .In AT5.1 the user will be shared the details of the ALU.

B15.4: Timing and Control unit

From B15.4 the Timing and Control unit section navigation is to be done. The content of the button will be shown in AT6.1

B15.5: Registers

Using this button user can proceed for the brief description of the register part .The content will be shown in AT7.1

B15.6: Data and Address Bus

This button is to navigate the user to the Data and Address Bus page from any of the scene of tutorial portion .The user will see the content of the page on the AT8.1 portion.

B15.7: Timing and Control Signal

For the Timing and Control Signal portion of the CBT B15.7 is used .In AT9.1 the user will be shared the details of the Timing and Control Signal.

B15.8: Fetch Operation

For the Fetch Operation portion of the CBT B15.8 is used .In AT10.1 the user will be shared the details of the Fetch Operation and user can use these buttons in any of the tutorial part

B15.9: Execute Operation

Using these button user can proceed for the brief description of the Execute Operation part .These buttons are available up to tutorial portion and content will be shown in AT11.1

B15.10: Machine Cycle

For the Machine Cycle portion of the CBT B15.10 is used .In AT12.1 the user will be shared the details of the Machine Cycle.

B15.11: Applications

This button will lead to the Applications portion of the CBT. Using B15.11 at AT13.1 the user will see the detailed Applications of 8085 A.

B15.12: Instruction 8085

Instruction 8085 button will take the user to the detailed Instruction 8085 portion.AT14.1 the brief description will be there .Sc15 has other navigation button inAT14.1 as well .User can go to any of the tutorial portion and once user reach this section after that user will get access of the Simulator ,Coding and Quiz section. Except this section user will not be allowed to go to the Simulator (Sc16), Coding (Sc17-Sc19) and Quiz (Sc20-24) section. In this section AT15.1 is for Data Transfer Group, AT15.2 is for Arithmetic Group, AT15.3 is for Logical Group, AT15.4 is for Branch Group and AT15.5 is for Stack, I/O and Machine Control Group. The user can navigate to the details for the instructions by tapping these buttons/Tabs.

B15.13: Simulator

This button will navigate the user to the simulator of the Intel 8085A Kit and will demonstrate how to work in that simulator . From 15.13 the user can go to this section .Once the button is used the user will be lead to the Simulator (Sc16) section.

B15.14: Coding

This button will navigate the user to the coding section is available. From 15.14 the user can go to this section. Once the button is used the user will be lead to Coding (Sc17-Sc19) section.

B15.15: Quiz

This button will navigate the user to the quiz section. From 15.15 the user can go to this section. Once the button is used the user will be lead to Quiz (Sc20-Sc24) section.

B15.16: Previous button

This button will navigate the user to the previous scene .It will lead the user to Sc.14

B15.17: Home Button

This button will navigate the user to the Content page scene (Sc. 3).

B15.18: Next Button

This button will navigate the user to the next scene. Sc.15 next button will lead to Sc.16

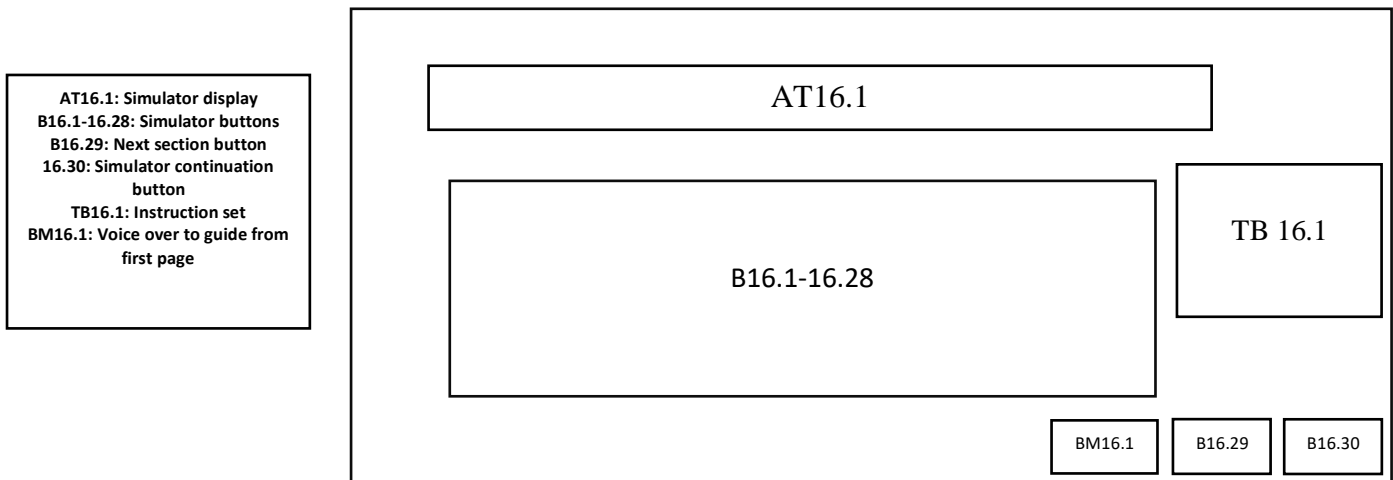
BM15.1: Voice Over

This BM15.1 button will trigger the audio file that is embedded to that scene .These button are start and stop button. User have to press the button manually at the beginning of the section. User have to start and stop the button manually. These voice over will guide the user regarding the section. It will navigate the user and will share the content of the scene as well.

AT15.1: Page Content

This button is for the description of the section .Using button the user can switch the tabs and the button that is pressed accordingly the content will be shown at AT3.1 to 14.1.These are animate text that are used using masking to give fade-in effect in text

/*Sc16:*/

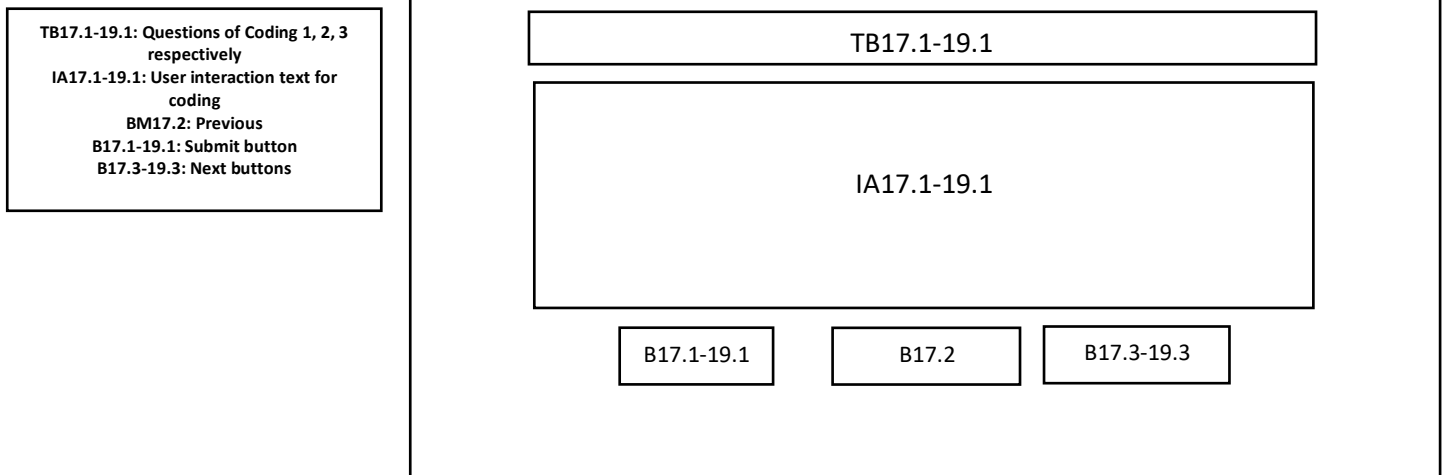


Sc.16 is for the simulator portion. Using AT16.1 Simulator display is done .Here the user will see the output after pressing the buttons after guided by Instruction set (TB16.1) .B16.1-16.28: Simulator buttons. These button names left to right is:

RESET-> for reset the kit,
VCT/INT-> Vector Interrupt. It generates hardware interrupt RST 7.5 via keypad,
SHIFT-> Provides second level commands to all keys,
C-> for C
D-> for D
E-> for E, F-> for F,
EXREG/SI-> Examine Register. It allows to see the values of different registers and Execute in Single Step Mode,
INS/DATA->Insert one or more data into memory,
DEL/DATA-> Delete a part of program
8H->8 and HL register pair use
9L->9 and HL register pair use
A->for A
B->for B
GO -> Execute the program
B.M-> Block Move. This helps to move a block of memory to any RAM area
REL/EXMEM-> Reallocates the program written for some memory area.
4/PCH->4 and Initializing PC register with H register content
5/PCL->5 and Initializing PC register with L register content
6/SPH->6 and Initializing SP register with H register content
7/SPL->7 and Initializing SP register with L register content
STRING/PRE-> Find a string of data lying at particular address/s
MEMC/NEXT -> Compare two blocks of memory for equality
FILL/. -> Fill some RAM area with constant values
0->for 0
1->for 1
2/SER->for 2 and Serial I/O signal
3-> for 3

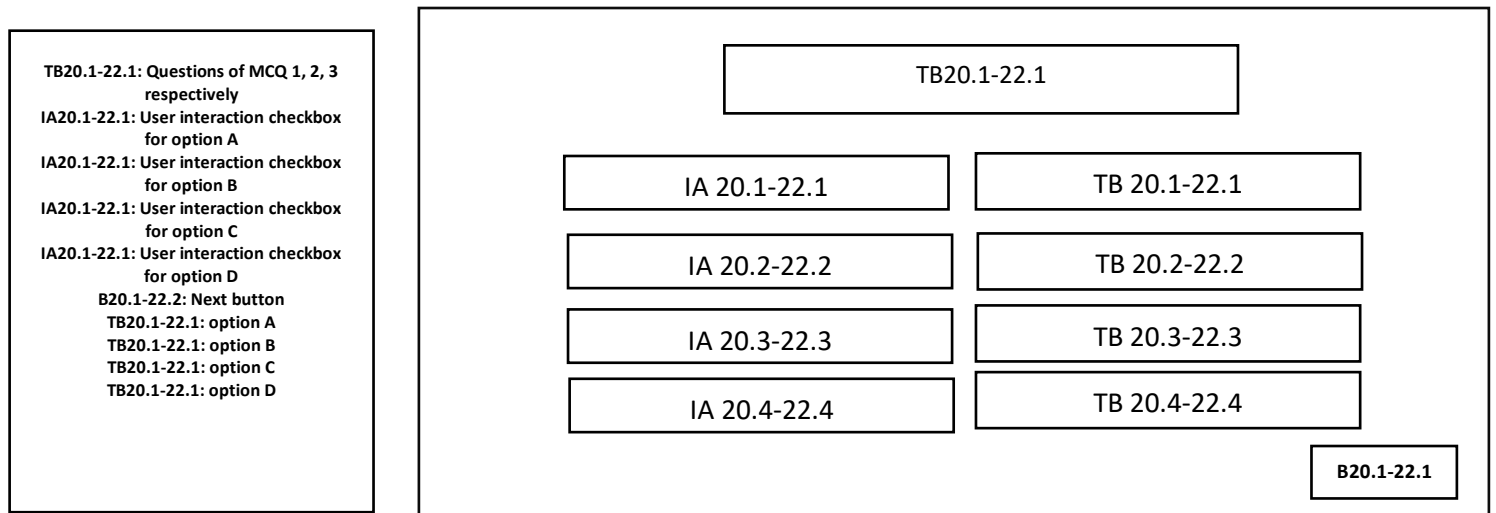
B16.29 is for Next section button. This will navigate the user to the Quiz section (Sc.17) and 16.30 is for Simulator continuation button i.e. this will lead the user to implement and use the 8085A kit .User will get familiar with the 8085A kit.TB16.1 is for the Instruction set where text will be highlighted to guide the user how to use the simulator.BM16.1: Voice over to guide from first page and will instruct regarding operation.

/*Sc17 to 19(all same):*/



TB17.1-19.1 is for the Questions of Coding 1, 2, 3 respectively .TB17.1->Question 1(Sc.17), TB18.1-> Question 2(Sc.18), TB19.1-> Question 3(Sc.19)
 IA17.1-19.1: User interaction text for coding (Sc.17, 18, 19)
 BM17.2: Previous button. This will navigate to the simulator section
 B17.1-19.1: Submit button for coding (Sc.17, 18, 19)
 B17.3-19.3: Next buttons for coding (Sc.17, 18, 19).B17.3 will take to next coding, same for B18.3 and 19.3 will take to MCQ question

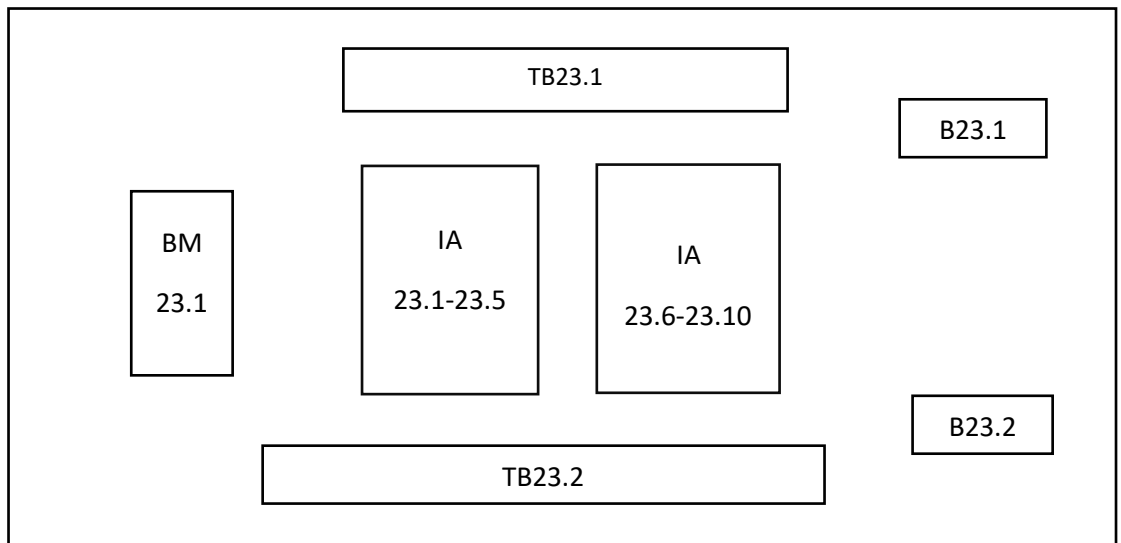
/*Sc20 to Sc22:*/



TB20.1-22.1 is used for the Questions of MCQ 1, 2, 3 respectively (Sc.20, 21, 22).IA20.1-22.1 is used for the user interaction checkbox for option A (from Sc20 to 22)IA20.1-22.1: User interaction checkbox for option B.IA20.1-22.1 is for the User interaction checkbox for option C and IA20.1-22.1 is for the user interaction checkbox for option D .B20.1-22.1 is for Next button.B20.1 is for the MCQ2 and B21.1 is for the MCQ2 and B22.1 will guide to match the following section.TB20.1-22.1 is for the option A with the probable answers .TB20.1-22.1 is option B with the probable answers.TB20.1-22.1 is option C with the probable answers.TB20.1-22.1 is option D with the probable answers.

/*Sc23:*/

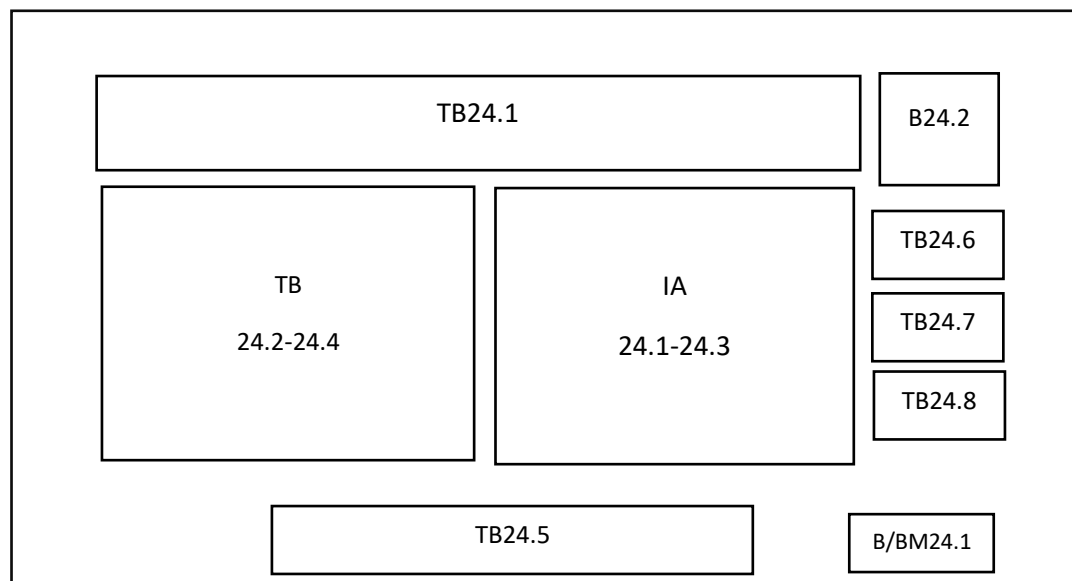
TB23.1: Text for question
BM23.1: Used for the instruction guide of the page
IA23.1-22.5: User interactive drag and drop box
IA23.6-23.10: User interactive drag and drop box
B23.1: Reset button
B23.2: Next button
TB23.2: For answer check



TB23.1 is used for the Text for question. User will get to know what to do in this section. The BM23.1 is used for the instruction guide of the page. This audio will direct the user to do drag and drop. IA23.1-22.5: User interactive drag and drop box. These options to be dragged to match with IA23.6-23.10. The IA23.6-23.10 is User interactive drag and drop box and these are used as target where user have to drag and drop the answer to match the answers. B23.1 is Reset button. This will help the user to start again from initial phase. B23.2: Next button TB23.2 will reflect the answer is correct or not. It will show "Keep Trying" until user corrects the matching.

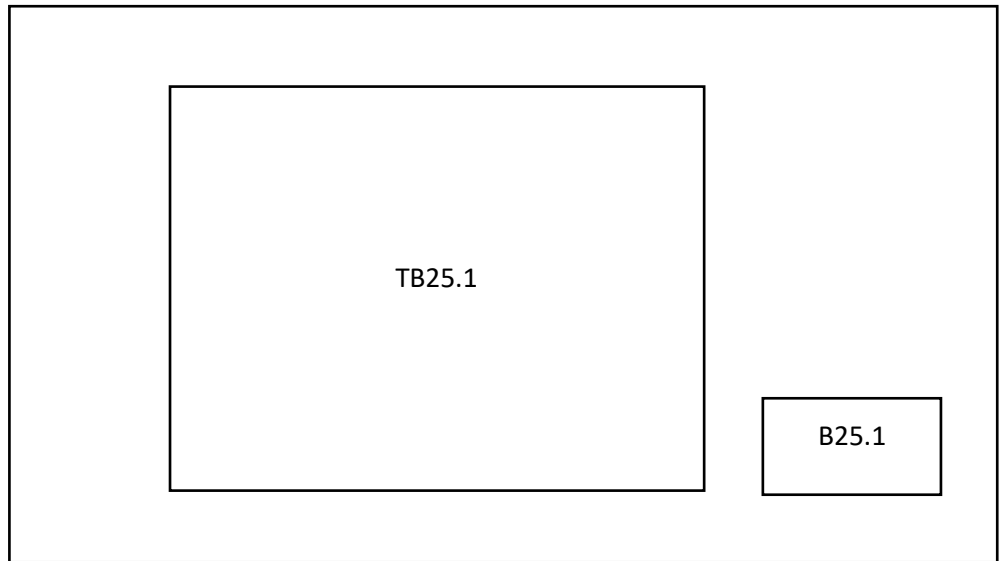
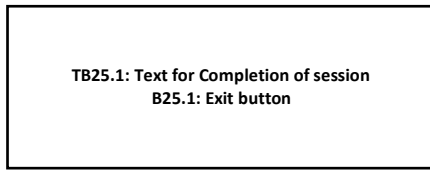
/*Sc24:*/

TB24.1: Text for question
TB24.2-24.4: Questions
IA24.1-24.3: combo box with options
B/BM 24.1: Next button and Voice over
TB24.5: Notification check
TB24.6: Question 1 Answer check
TB24.7: Question 2 Answer check
TB24.8: Question 3 Answer check
B24.2: All Answer check



TB24.1 is for text of question and TB24.2-24.4 are for the Questions. The IA24.1-24.3 is used for the combo box with options. The user have to choose among them and the correct answer will reflect there. B/BM 24.1: Next button to go to end page and the Voice over will start when the next button will be triggered. It will thank the user for completing the CBT on the next page. TB24.5 is for the notification that user is in last section of the CBT. TB24.6 is for Question 1 Answer check, TB24.7 is for Question 2 Answer check, TB24.8 is used for Question 3 Answer check. B24.2 is used for all Answer check at once. This TB24.9 will not be seen initially. This will appear after the user attempts the last question

/*Sc25:*/



The Storyboarding for this CBT is above mentioned .The Action scripts and media management catalogue that are used in next page

ACTION SCRIPT USED:

Here the action scripts are shared that is used for functioning the CBT. Page wise action scripts are shared .Action script saved location:

C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\actionScript

X----- Cover Page -----X

```
import flash.events.MouseEvent;

stop();

start.addEventListener(MouseEvent.CLICK, cover)

function cover(event: MouseEvent): void {
    gotoAndStop(1, "welcomePage");
    SoundMixer.stopAll();
}
```

X----- Welcome Page -----X

```
stop();

obj.addEventListener(MouseEvent.CLICK, fl_ClickToGoToAndStopAtFrame);

function fl_ClickToGoToAndStopAtFrame(event:MouseEvent):void
{
    gotoAndStop(5);
}

/*****/

TargetAud.addEventListener(MouseEvent.CLICK, f2_ClickToGoToAndStopAtFrame);

function f2_ClickToGoToAndStopAtFrame(event:MouseEvent):void
{
    gotoAndStop(10);
}

/*****/

SoftReq.addEventListener(MouseEvent.CLICK, f3_ClickToGoToAndStopAtFrame);

function f3_ClickToGoToAndStopAtFrame(event:MouseEvent):void
{
    gotoAndStop(15);
}
```

```

/*****/

Content.addEventListener(MouseEvent.CLICK, f4_ClickToGoToAndStopAtFrame);

function f4_ClickToGoToAndStopAtFrame(event:MouseEvent):void
{
    gotoAndStop(20);
}

/*****/

Reference.addEventListener(MouseEvent.CLICK, f5_ClickToGoToAndStopAtFrame);

function f5_ClickToGoToAndStopAtFrame(event:MouseEvent):void
{
    gotoAndStop(25);
}

/***** next btn *****/

import flash.events.MouseEvent;

stop();

nextScene2.addEventListener(MouseEvent.CLICK, welcome)

function welcome(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}

/*****/

```

/* Click to Play/Stop Sound
Clicking on the symbol instance plays the specified sound.
Clicking on the symbol instance a second time stops the sound.

Instructions:

1. Replace "http://www.helpexamples.com/flash/sound/song1.mp3" below with the desired URL address of your sound file. Keep the quotation marks ("").

*/

```

soundScene2.addEventListener(MouseEvent.CLICK, sound1);

var fl_SC1:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay1:Boolean = true;

```

```

function sound1(evt:MouseEvent):void
{
    if(fl_ToPlay1)
    {
        var s1:Sound= new welcomePageCorrected;
        fl_SC1 = s1.play();
    }
    else
    {
        fl_SC1.stop();
    }
    fl_ToPlay1 = !fl_ToPlay1;
}
----- Content Page -----

```

```

/***** next btn *****/

```

```

import flash.events.MouseEvent;

stop();

nextIntroBtn.addEventListener(MouseEvent.CLICK, content)

function content(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}

```

```

/***** WHOLE PAGE BUTTON NAVIGATION *****/

```

```

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

```

```

BtnIntro.addEventListener(MouseEvent.CLICK, BtnIntrouction)

```

```

function BtnIntrouction(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

```



```
BtnPinDiagram.addEventListener(MouseEvent.CLICK, BtnPinDiagrams)
```

```
function BtnPinDiagrams(event: MouseEvent): void {  
    gotoAndStop(1, "pinDiagram");  
    SoundMixer.stopAll();  
}  
/***** ARITHMATIC AND LOGIC UNIT *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnALU.addEventListener(MouseEvent.CLICK, BtnALUnit)
```

```
function BtnALUnit(event: MouseEvent): void {  
    gotoAndStop(1, "alu");  
    SoundMixer.stopAll();  
}  
/***** TIMING AND CONTROL UNIT *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTCL.addEventListener(MouseEvent.CLICK, BtnTCLunit)
```

```
function BtnTCLunit(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}  
/***** REGISTERS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnReg.addEventListener(MouseEvent.CLICK, BtnRegister)
```

```
function BtnRegister(event: MouseEvent): void {  
    gotoAndStop(1, "registers");  
    SoundMixer.stopAll();  
}  
/***** DATA AND ADDRESS BUS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnDataBus.addEventListener(MouseEvent.CLICK, BtnDataAddressBus)
```

```
function BtnDataAddressBus(event: MouseEvent): void {
```

```

        gotoAndStop(1, "dataAndAddressBus");
        SoundMixer.stopAll();
    }

    /***** TIMING AND CONTROL SIGNAL *****/
    import flash.events.MouseEvent;

    stop();

    BtnTACS.addEventListener(MouseEvent.CLICK, BtnTimingAndCtrlSg)

    function BtnTimingAndCtrlSg(event: MouseEvent): void {
        gotoAndStop(1, "timmingAndControlSignal");
        SoundMixer.stopAll();
    }
    /***** FETCH OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnFetch.addEventListener(MouseEvent.CLICK, fetchOptBtn)

    function fetchOptBtn(event: MouseEvent): void {
        gotoAndStop(1, "fetchOperation");
        SoundMixer.stopAll();
    }
    /***** EXECUTE OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnExec.addEventListener(MouseEvent.CLICK, BtnExecute)

    function BtnExecute(event: MouseEvent): void {
        gotoAndStop(1, "executeOperation");
        SoundMixer.stopAll();
    }
    /***** MACHINE CYCLE *****/
    import flash.events.MouseEvent;

    stop();

    BtnMchnCycl.addEventListener(MouseEvent.CLICK, BtnMachineCycle)

    function BtnMachineCycle(event: MouseEvent): void {
        gotoAndStop(1, "machineCycle");
        SoundMixer.stopAll();
    }

```

```

}
/***** APPLICATIONS *****/
import flash.events.MouseEvent;

stop();

BtnApp.addEventListener(MouseEvent.CLICK, BtnApplication)

function BtnApplication(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

BtnInstruc.addEventListener(MouseEvent.CLICK, BtnInstruction)

function BtnInstruction(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}
/*****
*****/

SoundContent.addEventListener(MouseEvent.CLICK, sound2);

var fl_SC2:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay2:Boolean = true;

function sound2(evt:MouseEvent):void
{
    if(fl_ToPlay2)
    {
        var s2:Sound = new contentAudio;
        fl_SC2 = s2.play();
    }
    else
    {
        fl_SC2.stop();
    }
    fl_ToPlay2 = !fl_ToPlay2;
}
/*****
*****/

```

X----- Introduction -----X

```

/***** Home Button *****/

import flash.events.MouseEvent;

stop();

Home45.addEventListener(MouseEvent.CLICK, home450)

function home450(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
/***** Next button *****/

import flash.events.MouseEvent;

stop();

nextIntroBtn.addEventListener(MouseEvent.CLICK, nextIntro)

function nextIntro(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}

/***** Back button *****/

import flash.events.MouseEvent;

stop();

prevIntroBtn.addEventListener(MouseEvent.CLICK, prevIntro)

function prevIntro(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
/*****
*****/

/***** WHOLE PAGE BUTTON NAVIGATION *****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnIntro1.addEventListener(MouseEvent.CLICK, BtnIntro1)
```

```
function BtnIntro1(event: MouseEvent): void {  
    gotoAndStop(1, "introduction");  
    SoundMixer.stopAll();  
}
```

```
/****** PIN DIAGRAM *****/  
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnPinDiagram1.addEventListener(MouseEvent.CLICK, BtnPinDiagrams1)
```

```
function BtnPinDiagrams1(event: MouseEvent): void {  
    gotoAndStop(1, "pinDiagram");  
    SoundMixer.stopAll();  
}
```

```
/****** ARITHMETIC AND LOGIC UNIT *****/  
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnALU1.addEventListener(MouseEvent.CLICK, BtnALUnit1)
```

```
function BtnALUnit1(event: MouseEvent): void {  
    gotoAndStop(1, "alu");  
    SoundMixer.stopAll();  
}
```

```
/****** TIMING AND CONTROL UNIT * *****/  
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnTCL1.addEventListener(MouseEvent.CLICK, BtnTCLUnit1)
```

```
function BtnTCLUnit1(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}
```

```
/****** REGISTERS *****/  
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnReg1.addEventListener(MouseEvent.CLICK, BtnRegister1)
```

```
function BtnRegister1(event: MouseEvent): void {  
    gotoAndStop(1, "registers");  
    SoundMixer.stopAll();  
}  
/***** DATA AND ADDRESS BUS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnDataBus1.addEventListener(MouseEvent.CLICK, BtnDataAddressBus1)
```

```
function BtnDataAddressBus1(event: MouseEvent): void {  
    gotoAndStop(1, "dataAndAddressBus");  
    SoundMixer.stopAll();  
}  
  
/***** TIMING AND CONTROL SIGNAL *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTACS1.addEventListener(MouseEvent.CLICK, BtntimingAndCtrlSg1)
```

```
function BtntimingAndCtrlSg1(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
    SoundMixer.stopAll();  
}  
/***** FETCH OPERATION *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnFetch1.addEventListener(MouseEvent.CLICK, fetchOptBtn1)
```

```
function fetchOptBtn1(event: MouseEvent): void {  
    gotoAndStop(1, "fetchOperation");  
    SoundMixer.stopAll();  
}  
/***** EXECUTE OPERATION *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnExec1.addEventListener(MouseEvent.CLICK, BtnExecute1)
```

```

function BtnExecute1(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}
/***** MACHINE CYCLE *****/
import flash.events.MouseEvent;

stop();

BtnMchnCycl1.addEventListener(MouseEvent.CLICK, BtnMachineCycle1)

function BtnMachineCycle1(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
/***** APPLICATIONS *****/
import flash.events.MouseEvent;

stop();

BtnApp1.addEventListener(MouseEvent.CLICK, BtnApplication1)

function BtnApplication1(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

BtnInstruc1.addEventListener(MouseEvent.CLICK, BtnInstruction1)

function BtnInstruction1(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}
/*****

/*****

IntroSound.addEventListener(MouseEvent.CLICK, sound3);

var fl_SC3:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay3:Boolean = true;

```

```

function sound3(evt:MouseEvent):void
{
    if(fl_ToPlay3)
    {
        var s3:Sound = new introduction;
        fl_SC3 = s3.play();
    }
    else
    {
        fl_SC3.stop();
    }
    fl_ToPlay3 = !fl_ToPlay3;
}
/*****
----- Pin Diagram -----
*****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro2.addEventListener(MouseEvent.CLICK, BtnIntrouction2)

function BtnIntrouction2(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram2.addEventListener(MouseEvent.CLICK, BtnPinDiagrams2)

function BtnPinDiagrams2(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/***** ARITHMATIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU2.addEventListener(MouseEvent.CLICK, BtnALUnit2)

```



```

function BtnALUnit2(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/***** TIMING AND CONTROL UNIT *****/
import flash.events.MouseEvent;

stop();

BtnTCL2.addEventListener(MouseEvent.CLICK, BtnTCLUnit2)

function BtnTCLUnit2(event: MouseEvent): void {
    gotoAndStop(1, "timingAndControlUnit");
    SoundMixer.stopAll();
}
/***** REGISTERS *****/
import flash.events.MouseEvent;

stop();

BtnReg2.addEventListener(MouseEvent.CLICK, BtnRegister2)

function BtnRegister2(event: MouseEvent): void {
    gotoAndStop(1, "registers");
    SoundMixer.stopAll();
}
/***** DATA AND ADDRESS BUS *****/
import flash.events.MouseEvent;

stop();

BtnDataBus2.addEventListener(MouseEvent.CLICK, BtnDataAddressBus2)

function BtnDataAddressBus2(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
    SoundMixer.stopAll();
}

/***** TIMING AND CONTROL SIGNAL *****/
import flash.events.MouseEvent;

stop();

BtnTACS2.addEventListener(MouseEvent.CLICK, BtnTimingAndCtrlSg2)

function BtnTimingAndCtrlSg2(event: MouseEvent): void {

```

```

        gotoAndStop(1, "timmingAndControlSignal");
        SoundMixer.stopAll();
    }
    /***** FETCH OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnFetch2.addEventListener(MouseEvent.CLICK, fetchOptBtn2)

    function fetchOptBtn2(event: MouseEvent): void {
        gotoAndStop(1, "fetchOperation");
        SoundMixer.stopAll();
    }
    /***** EXECUTE OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnExec2.addEventListener(MouseEvent.CLICK, BtnExecute2)

    function BtnExecute2(event: MouseEvent): void {
        gotoAndStop(1, "executeOperation");
        SoundMixer.stopAll();
    }
    /***** MACHINE CYCLE *****/
    import flash.events.MouseEvent;

    stop();

    BtnMchnCycl2.addEventListener(MouseEvent.CLICK, BtnMachineCycle2)

    function BtnMachineCycle2(event: MouseEvent): void {
        gotoAndStop(1, "machineCycle");
        SoundMixer.stopAll();
    }
    /***** APPLICATIONS *****/
    import flash.events.MouseEvent;

    stop();

    BtnApp2.addEventListener(MouseEvent.CLICK, BtnApplication2)

    function BtnApplication2(event: MouseEvent): void {
        gotoAndStop(1, "applications");
        SoundMixer.stopAll();
    }

```

```

/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

BtnInstruc2.addEventListener(MouseEvent.CLICK, BtnInstruction2)

function BtnInstruction2(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}
/*****/

/***** Next button *****/

import flash.events.MouseEvent;

stop();

nextPinDiaBtn.addEventListener(MouseEvent.CLICK, pinDia)

function pinDia(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}

/***** Home Button *****/

import flash.events.MouseEvent;

stop();

Home50.addEventListener(MouseEvent.CLICK, Home50pinDia)

function Home50pinDia(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}

/***** Back button *****/

import flash.events.MouseEvent;

stop();

prevPinDiaBtn.addEventListener(MouseEvent.CLICK, prevPinDia)

function prevPinDia(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}

```

```

}
/*****

/*****/

PinDiagramSound.addEventListener(MouseEvent.CLICK, sound4);

var fl_SC4:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay4:Boolean = true;

function sound4(evt:MouseEvent):void
{
    if(fl_ToPlay4)
    {
        var s4:Sound = new pinDiagram;
        fl_SC4 = s4.play();
    }
    else
    {
        fl_SC4.stop();
    }
    fl_ToPlay4 = !fl_ToPlay4;
}
/*****
----- ALU -----
/***** Next button *****/

import flash.events.MouseEvent;

stop();

nextALUBtn.addEventListener(MouseEvent.CLICK, ALU)

function ALU(event: MouseEvent): void {
    gotoAndStop(1, "timingAndControlUnit");
    SoundMixer.stopAll();
}

/***** Home Button *****/

import flash.events.MouseEvent;

stop();

```

```
function home12345(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
/***** Back button *****/
```

```
import flash.events.MouseEvent;

stop();
```

```
prevALUBtn.addEventListener(MouseEvent.CLICK, prevInstr1200)
```

```
function prevInstr1200(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/*****
/***** WHOLE PAGE BUTTON NAVIGATION
*****/
```

```

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

```

```
BtnIntro3.addEventListener(MouseEvent.CLICK, BtnIntro3Click);
```

```
function BtnIntroction3(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();
```

```
BtnPinDiagram3.addEventListener(MouseEvent.CLICK, BtnPinDiagrams3)
```

```
function BtnPinDiagrams3(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
}
/***** ARITHMETIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();
```

```
BtnALU3.addEventListener(MouseEvent.CLICK, BtnALUnit3)
```

```
function BtnALUnit3(event: MouseEvent): void {  
    gotoAndStop(1, "alu");  
}  
/***** TIMING AND CONTROL UNIT *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTCL3.addEventListener(MouseEvent.CLICK, BtnTCLUnit3)
```

```
function BtnTCLUnit3(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
}  
/***** REGISTERS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnReg3.addEventListener(MouseEvent.CLICK, BtnRegister3)
```

```
function BtnRegister3(event: MouseEvent): void {  
    gotoAndStop(1, "registers");  
}  
/***** DATA AND ADDRESS BUS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnDataBus3.addEventListener(MouseEvent.CLICK, BtnDataAddressBus3)
```

```
function BtnDataAddressBus3(event: MouseEvent): void {  
    gotoAndStop(1, "dataAndAddressBus");  
}  
  
/***** TIMING AND CONTROL SIGNAL *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTACS3.addEventListener(MouseEvent.CLICK, BtntimingAndCtrlSg3)
```

```
function BtntimingAndCtrlSg3(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
}
```

```
/****** FETCH OPERATION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnFetch3.addEventListener(MouseEvent.CLICK, fetchOptBtn3)
```

```
function fetchOptBtn3(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
}
```

```
/****** EXECUTE OPERATION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnExec3.addEventListener(MouseEvent.CLICK, BtnExecute3)
```

```
function BtnExecute3(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
}
```

```
/****** MACHINE CYCLE *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnMchnCycl3.addEventListener(MouseEvent.CLICK, BtnMachineCycle3)
```

```
function BtnMachineCycle3(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
}
```

```
/****** APPLICATIONS *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnApp3.addEventListener(MouseEvent.CLICK, BtnApplication3)
```

```
function BtnApplication3(event: MouseEvent): void {
    gotoAndStop(1, "applications");
}
```

```
/****** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnInstruc3.addEventListener(MouseEvent.CLICK, BtnInstruction3)
```

```

function BtnInstruction3(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
}
/*****
/*****

```

```

ALUSound.addEventListener(MouseEvent.CLICK, sound5);

```

```

var fl_SC5:SoundChannel;

```

```

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay5:Boolean = true;

```

```

function sound5(evt:MouseEvent):void
{
    if(fl_ToPlay5)
    {
        var s5:Sound = new alu;
        fl_SC5 = s5.play();
    }
    else
    {
        fl_SC5.stop();
    }
    fl_ToPlay5 = !fl_ToPlay5;
}
/*****

```

X-----Timing and Control unit-----X

```

/***** Home Button *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

Home60.addEventListener(MouseEvent.CLICK, home600)

```

```

function home600(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}

```

```

/***** Next button *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

nextTcuBtn.addEventListener(MouseEvent.CLICK, nextTcu)

```



```

function nextTcu(event: MouseEvent): void {
    gotoAndStop(1, "registers");
    SoundMixer.stopAll();
}

/***** Back button *****/

import flash.events.MouseEvent;

stop();

prevTcuBtn1.addEventListener(MouseEvent.CLICK, prevTcu)

function prevTcu(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/*****/
/***** WHOLE PAGE BUTTON NAVIGATION *****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro4.addEventListener(MouseEvent.CLICK, BtnIntro4)

function BtnIntro4(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/*****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram4.addEventListener(MouseEvent.CLICK, BtnPinDiagrams4)

function BtnPinDiagrams4(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/*****/
import flash.events.MouseEvent;

stop();

```

```
BtnALU4.addEventListener(MouseEvent.CLICK, BtnALUnit4)
```

```
function BtnALUnit4(event: MouseEvent): void {  
    gotoAndStop(1, "alu");  
    SoundMixer.stopAll();  
}  
/***** TIMING AND CONTROL UNIT * *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTCL4.addEventListener(MouseEvent.CLICK, BtnTCLUnit4)
```

```
function BtnTCLUnit4(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}  
/***** REGISTERS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnReg4.addEventListener(MouseEvent.CLICK, BtnRegister4)
```

```
function BtnRegister4(event: MouseEvent): void {  
    SoundMixer.stopAll();  
    gotoAndStop(1, "registers");  
    SoundMixer.stopAll();  
}  
/***** DATA AND ADDRESS BUS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnDataBus4.addEventListener(MouseEvent.CLICK, BtnDataAddressBus4)
```

```
function BtnDataAddressBus4(event: MouseEvent): void {  
    gotoAndStop(1, "dataAndAddressBus");  
    SoundMixer.stopAll();  
}  
  
/***** TIMING AND CONTROL SIGNAL *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTACS4.addEventListener(MouseEvent.CLICK, BtnTimingAndCtrlSg4)
```

```
function BtnTimingAndCtrlSg4(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
    SoundMixer.stopAll();  
}  
/***** FETCH OPERATION *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnFetch4.addEventListener(MouseEvent.CLICK, fetchOptBtn4)
```

```
function fetchOptBtn4(event: MouseEvent): void {  
    gotoAndStop(1, "fetchOperation");  
    SoundMixer.stopAll();  
}  
/***** EXECUTE OPERATION *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnExec4.addEventListener(MouseEvent.CLICK, BtnExecute4)
```

```
function BtnExecute4(event: MouseEvent): void {  
    gotoAndStop(1, "executeOperation");  
    SoundMixer.stopAll();  
}  
/***** MACHINE CYCLE *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnMchnCycl4.addEventListener(MouseEvent.CLICK, BtnMachineCycle4)
```

```
function BtnMachineCycle4(event: MouseEvent): void {  
    gotoAndStop(1, "machineCycle");  
    SoundMixer.stopAll();  
}  
/***** APPLICATIONS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnApp4.addEventListener(MouseEvent.CLICK, BtnApplication4)
```

```

function BtnApplication4(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

```

```

BtnInstruc4.addEventListener(MouseEvent.CLICK, BtnInstruction4)

```

```

function BtnInstruction4(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}

```

```

/*****/

```

```

TCUSound.addEventListener(MouseEvent.CLICK, sound6);

```

```

var fl_SC6:SoundChannel;

```

```

//This variable keeps track of whether you want to play or stop the sound

```

```

var fl_ToPlay6:Boolean = true;

```

```

function sound6(evt:MouseEvent):void
{
    if(fl_ToPlay6)
    {
        var s6:Sound = new TimingandControlUnit;
        fl_SC6 = s6.play();
    }
    else
    {
        fl_SC6.stop();
    }
    fl_ToPlay6 = !fl_ToPlay6;
}

```

```

/*****/

```

X----- Registers -----X

```

/***** next btn *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

nextRegBtn.addEventListener(MouseEvent.CLICK, registers12)

```

```

function registers12(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
}

```

```

        SoundMixer.stopAll();
    }

/***** Home Button *****/

import flash.events.MouseEvent;

stop();

Home65.addEventListener(MouseEvent.CLICK, homeReg)

function homeReg(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}

/***** Back button *****/

import flash.events.MouseEvent;

stop();

prevRegBtn.addEventListener(MouseEvent.CLICK, prevReg)

function prevReg(event: MouseEvent): void {
    gotoAndStop(1, "timingAndControlUnit");
    SoundMixer.stopAll();
}
/*****//*****/
**** WHOLE PAGE BUTTON NAVIGATION *****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro5.addEventListener(MouseEvent.CLICK, BtnIntro5)

function BtnIntro5(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram5.addEventListener(MouseEvent.CLICK, BtnPinDiagrams5)

function BtnPinDiagrams5(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}

```

```

}
/***** ARITHMETIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU5.addEventListener(MouseEvent.CLICK, BtnALUnit5)

function BtnALUnit5(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/***** TIMING AND CONTROL UNIT *****/
import flash.events.MouseEvent;

stop();

BtnTCL5.addEventListener(MouseEvent.CLICK, BtnTCLunit5)

function BtnTCLunit5(event: MouseEvent): void {
    gotoAndStop(1, "timingAndControlUnit");
    SoundMixer.stopAll();
}
/***** REGISTERS *****/
import flash.events.MouseEvent;

stop();

BtnReg5.addEventListener(MouseEvent.CLICK, BtnRegister5)

function BtnRegister5(event: MouseEvent): void {
    gotoAndStop(1, "registers");
    SoundMixer.stopAll();
}
/***** DATA AND ADDRESS BUS *****/
import flash.events.MouseEvent;

stop();

BtnDataBus5.addEventListener(MouseEvent.CLICK, BtnDataAddressBus5)

function BtnDataAddressBus5(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
    SoundMixer.stopAll();
}

/***** TIMING AND CONTROL SIGNAL *****/
import flash.events.MouseEvent;

stop();

BtnTACS5.addEventListener(MouseEvent.CLICK, BntimingAndCtrlSg5)

```

```

function BtnTimingAndCtrlSg5(event: MouseEvent): void {
    gotoAndStop(1, "timmingAndControlSignal");
    SoundMixer.stopAll();
}
/***** FETCH OPERATION *****/
import flash.events.MouseEvent;

stop();

BtnFetch5.addEventListener(MouseEvent.CLICK, fetchOptBtn5)

function fetchOptBtn5(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}
/***** EXECUTE OPERATION *****/
import flash.events.MouseEvent;

stop();

BtnExec5.addEventListener(MouseEvent.CLICK, BtnExecute5)

function BtnExecute5(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}
/***** MACHINE CYCLE *****/
import flash.events.MouseEvent;

stop();

BtnMchnCycl5.addEventListener(MouseEvent.CLICK, BtnMachineCycle5)

function BtnMachineCycle5(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
/***** APPLICATIONS *****/
import flash.events.MouseEvent;

stop();

BtnApp5.addEventListener(MouseEvent.CLICK, BtnApplication5)

function BtnApplication5(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

```

```
BtnInstruc5.addEventListener(MouseEvent.CLICK, BtnInstruction5)
```

```
function BtnInstruction5(event: MouseEvent): void {  
    gotoAndStop(1, "instruction8085");  
    SoundMixer.stopAll();  
}
```

```
/******  
/
```

```
RegSound.addEventListener(MouseEvent.CLICK, sound7);
```

```
var fl_SC7:SoundChannel;
```

```
//This variable keeps track of whether you want to play or stop the sound  
var fl_ToPlay7:Boolean = true;
```

```
function sound7(evt:MouseEvent):void  
{
```

```
    if(fl_ToPlay7)
```

```
    {
```

```
        var s7:Sound = new registers;
```

```
        fl_SC7 = s7.play();
```

```
    }
```

```
    else
```

```
    {
```

```
        fl_SC7.stop();
```

```
    }
```

```
    fl_ToPlay7 = !fl_ToPlay7;
```

```
}
```

```
/******  
/
```

X----- Data and Address Bus -----X

```
/****** Next Btn *****  
/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
nextDataBusBtn.addEventListener(MouseEvent.CLICK, dataBus)
```

```
function dataBus(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
    SoundMixer.stopAll();  
}
```

```
}
```

```
/****** Previous Btn *****  
/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
prevDataBusBtn.addEventListener(MouseEvent.CLICK, dataBus2)
```

```
function dataBus2(event: MouseEvent): void {
```



```

        gotoAndStop(1, "registers");
        SoundMixer.stopAll();
    }

    /***** Home Btn *****/

import flash.events.MouseEvent;

stop();

Home70.addEventListener(MouseEvent.CLICK, dataBus3)

function dataBus3(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
/***** WHOLE PAGE BUTTON NAVIGATION *****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro6.addEventListener(MouseEvent.CLICK, BtnIntro6)

function BtnIntro6(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram6.addEventListener(MouseEvent.CLICK, BtnPinDiagrams6)

function BtnPinDiagrams6(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/***** ARITHMETIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU6.addEventListener(MouseEvent.CLICK, BtnALUnit6)

function BtnALUnit6(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/***** TIMING AND CONTROL UNIT *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnTCL6.addEventListener(MouseEvent.CLICK, BtnTCLUnit6)
```

```
function BtnTCLUnit6(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}
```

```
/****** REGISTERS *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnReg6.addEventListener(MouseEvent.CLICK, BtnRegister6)
```

```
function BtnRegister6(event: MouseEvent): void {  
    gotoAndStop(1, "registers");  
    SoundMixer.stopAll();  
}
```

```
/****** DATA AND ADDRESS BUS *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnDataBus6.addEventListener(MouseEvent.CLICK, BtnDataAddressBus6)
```

```
function BtnDataAddressBus6(event: MouseEvent): void {  
    gotoAndStop(1, "dataAndAddressBus");  
    SoundMixer.stopAll();  
}
```

```
/****** TIMING AND CONTROL SIGNAL *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnTACS6.addEventListener(MouseEvent.CLICK, BtntimingAndCtrlSg6)
```

```
function BtntimingAndCtrlSg6(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
    SoundMixer.stopAll();  
}
```

```
/****** FETCH OPERATION *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnFetch6.addEventListener(MouseEvent.CLICK, fetchOptBtn6)
```

```
function fetchOptBtn6(event: MouseEvent): void {
```

```

        gotoAndStop(1, "fetchOperation");
        SoundMixer.stopAll();
    }
    /***** EXECUTE OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnExec6.addEventListener(MouseEvent.CLICK, BtnExecute6)

    function BtnExecute6(event: MouseEvent): void {
        gotoAndStop(1, "executeOperation");
        SoundMixer.stopAll();
    }
    /***** MACHINE CYCLE *****/
    import flash.events.MouseEvent;

    stop();

    BtnMchnCycl6.addEventListener(MouseEvent.CLICK, BtnMachineCycle6)

    function BtnMachineCycle6(event: MouseEvent): void {
        gotoAndStop(1, "machineCycle");
        SoundMixer.stopAll();
    }
    /***** APPLICATIONS *****/
    import flash.events.MouseEvent;

    stop();

    BtnApp6.addEventListener(MouseEvent.CLICK, BtnApplication6)

    function BtnApplication6(event: MouseEvent): void {
        gotoAndStop(1, "applications");
        SoundMixer.stopAll();
    }
    /***** 8085 INSTRUCTION *****/
    import flash.events.MouseEvent;

    stop();

    BtnInstruc6.addEventListener(MouseEvent.CLICK, BtnInstruction6)

    function BtnInstruction6(event: MouseEvent): void {
        gotoAndStop(1, "instruction8085");
        SoundMixer.stopAll();
    }
    /*****

    DataAddressBusSound.addEventListener(MouseEvent.CLICK, sound8);

    var fl_SC8:SoundChannel;

```

```
//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay8:Boolean = true;
```

```
function sound8(evt:MouseEvent):void
```

```
{
    if(fl_ToPlay8)
    {
        var s8:Sound = new dataandaddressbus;
        fl_SC8 = s8.play();
    }
    else
    {
        fl_SC8.stop();
    }
    fl_ToPlay8 = !fl_ToPlay8;
}
```

```
/******
```

X----- Timing and Control Signal -----X

```
/****** Home Button *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Home75.addEventListener(MouseEvent.CLICK, hometime)
```

```
function hometime(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
```

```
/****** Next button *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
nextTimingBtn.addEventListener(MouseEvent.CLICK, nextTiming)
```

```
function nextTiming(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}
```

```
/****** Back button *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
prevTimingBtn.addEventListener(MouseEvent.CLICK, prevTiming)
```

```

function prevTimming(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
    SoundMixer.stopAll();
}
/*****

/***** WHOLE PAGE BUTTON NAVIGATION
*****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro7.addEventListener(MouseEvent.CLICK, BtnIntrouction7)

function BtnIntrouction7(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram7.addEventListener(MouseEvent.CLICK, BtnPinDiagrams7)

function BtnPinDiagrams7(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/***** ARITHMATIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU7.addEventListener(MouseEvent.CLICK, BtnALUnit7)

function BtnALUnit7(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/***** TIMING AND CONTROL UNIT * *****/
import flash.events.MouseEvent;

stop();

```

```
BtnTCL7.addEventListener(MouseEvent.CLICK, BtnTCLUnit7)
```

```
function BtnTCLUnit7(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}  
/***** REGISTERS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnReg7.addEventListener(MouseEvent.CLICK, BtnRegister7)
```

```
function BtnRegister7(event: MouseEvent): void {  
    gotoAndStop(1, "registers");  
    SoundMixer.stopAll();  
}  
/***** DATA AND ADDRESS BUS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnDataBus7.addEventListener(MouseEvent.CLICK, BtnDataAddressBus7)
```

```
function BtnDataAddressBus7(event: MouseEvent): void {  
    gotoAndStop(1, "dataAndAddressBus");  
    SoundMixer.stopAll();  
}  
  
/***** TIMING AND CONTROL SIGNAL *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTACS7.addEventListener(MouseEvent.CLICK, BntimingAndCtrlSg7)
```

```
function BntimingAndCtrlSg7(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
    SoundMixer.stopAll();  
}  
/***** FETCH OPERATION *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnFetch7.addEventListener(MouseEvent.CLICK, fetchOptBtn7)
```

```

function fetchOptBtn7(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}
/***** EXECUTE OPERATION *****/
import flash.events.MouseEvent;

stop();

```

```

BtnExec7.addEventListener(MouseEvent.CLICK, BtnExecute7)

```

```

function BtnExecute7(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}
/***** MACHINE CYCLE *****/
import flash.events.MouseEvent;

stop();

```

```

BtnMchnCycl7.addEventListener(MouseEvent.CLICK, BtnMachineCycle7)

```

```

function BtnMachineCycle7(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
/***** APPLICATIONS *****/
import flash.events.MouseEvent;

stop();

```

```

BtnApp7.addEventListener(MouseEvent.CLICK, BtnApplication7)

```

```

function BtnApplication7(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

```

```

BtnInstruc7.addEventListener(MouseEvent.CLICK, BtnInstruction7)

```

```

function BtnInstruction7(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
}

```

```

        SoundMixer.stopAll();
    }
    /*****
        *****/

```

```

TimingAndControlSignalSound.addEventListener(MouseEvent.CLICK, sound9);

```

```

var fl_SC9:SoundChannel;

```

```

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay9:Boolean = true;

```

```

function sound9(evt:MouseEvent):void
{
    if(fl_ToPlay9)
    {
        var s9:Sound = new timingcontrolsignal;
        fl_SC9 = s9.play();
    }
    else
    {
        fl_SC9.stop();
    }
    fl_ToPlay9 = !fl_ToPlay9;
}
/*****

```

X----- Fetch Operation -----X

```

/***** Home Button *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

Home80.addEventListener(MouseEvent.CLICK, home4)

```

```

function home4(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}

```

```

/***** Next button *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

nextFetchBtn.addEventListener(MouseEvent.CLICK, nextFetch)

```

```

function nextFetch(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
}

```



```

        SoundMixer.stopAll();
    }

    /***** Back button *****/

import flash.events.MouseEvent;

stop();

prevFetchBtn.addEventListener(MouseEvent.CLICK, prevFetch)

function prevFetch(event: MouseEvent): void {
    gotoAndStop(1, "timmingAndControlSignal");
    SoundMixer.stopAll();
}

/***** WHOLE PAGE BUTTON NAVIGATION *****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro8.addEventListener(MouseEvent.CLICK, BtnIntro8)

function BtnIntro8(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}

/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram8.addEventListener(MouseEvent.CLICK, BtnPinDiagrams8)

function BtnPinDiagrams8(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}

/***** ARITHMETIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU8.addEventListener(MouseEvent.CLICK, BtnALUnit8)

```

```

function BtnALUnit8(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/***** TIMING AND CONTROL UNIT *****/
import flash.events.MouseEvent;

stop();

BtnTCL8.addEventListener(MouseEvent.CLICK, BtnTCLUnit8)

function BtnTCLUnit8(event: MouseEvent): void {
    gotoAndStop(1, "timingAndControlUnit");
    SoundMixer.stopAll();
}
/***** REGISTERS *****/
import flash.events.MouseEvent;

stop();

BtnReg8.addEventListener(MouseEvent.CLICK, BtnRegister8)

function BtnRegister8(event: MouseEvent): void {
    gotoAndStop(1, "registers");
    SoundMixer.stopAll();
}
/***** DATA AND ADDRESS BUS *****/
import flash.events.MouseEvent;

stop();

BtnDataBus8.addEventListener(MouseEvent.CLICK, BtnDataAddressBus8)

function BtnDataAddressBus8(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
    SoundMixer.stopAll();
}

/***** TIMING AND CONTROL SIGNAL *****/
import flash.events.MouseEvent;

stop();

BtnTACS8.addEventListener(MouseEvent.CLICK, BtnTimingAndCtrlSg8)

function BtnTimingAndCtrlSg8(event: MouseEvent): void {

```

```

        gotoAndStop(1, "timmingAndControlSignal");
        SoundMixer.stopAll();
    }
    /***** FETCH OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnFetch8.addEventListener(MouseEvent.CLICK, fetchOptBtn8)

    function fetchOptBtn8(event: MouseEvent): void {
        gotoAndStop(1, "fetchOperation");
        SoundMixer.stopAll();
    }
    /***** EXECUTE OPERATION *****/
    import flash.events.MouseEvent;

    stop();

    BtnExec8.addEventListener(MouseEvent.CLICK, BtnExecute8)

    function BtnExecute8(event: MouseEvent): void {
        gotoAndStop(1, "executeOperation");
        SoundMixer.stopAll();
    }
    /***** MACHINE CYCLE *****/
    import flash.events.MouseEvent;

    stop();

    BtnMchnCycl8.addEventListener(MouseEvent.CLICK, BtnMachineCycle8)

    function BtnMachineCycle8(event: MouseEvent): void {
        gotoAndStop(1, "machineCycle");
        SoundMixer.stopAll();
    }
    /***** APPLICATIONS *****/
    import flash.events.MouseEvent;

    stop();

    BtnApp8.addEventListener(MouseEvent.CLICK, BtnApplication8)

    function BtnApplication8(event: MouseEvent): void {
        gotoAndStop(1, "applications");
        SoundMixer.stopAll();
    }

```

```

/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

BtnInstruc8.addEventListener(MouseEvent.CLICK, BtnInstruction8)

function BtnInstruction8(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}
/*****/

```

```

FetchSound.addEventListener(MouseEvent.CLICK, sound10);

var fl_SC10:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay10:Boolean = true;

```

```

function sound10(evt:MouseEvent):void
{
    if(fl_ToPlay10)
    {
        var s10:Sound = new fetchOperation;
        fl_SC10 = s10.play();
    }
    else
    {
        fl_SC10.stop();
    }
    fl_ToPlay10 = !fl_ToPlay10;
}
/*****/

```

X----- Execute Operation -----X

```

/***** WHOLE PAGE BUTTON NAVIGATION *****/

```

```

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro9.addEventListener(MouseEvent.CLICK, BtnIntro9)

function BtnIntro9(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
}

```

```

        SoundMixer.stopAll();
    }
    /***** PIN DIAGRAM *****/
    import flash.events.MouseEvent;

    stop();

    BtnPinDiagram9.addEventListener(MouseEvent.CLICK, BtnPinDiagrams9)

    function BtnPinDiagrams9(event: MouseEvent): void {
        gotoAndStop(1, "pinDiagram");
        SoundMixer.stopAll();
    }
    /***** ARITHMETIC AND LOGIC UNIT *****/
    import flash.events.MouseEvent;

    stop();

    BtnALU9.addEventListener(MouseEvent.CLICK, BtnALUnit9)

    function BtnALUnit9(event: MouseEvent): void {
        gotoAndStop(1, "alu");
        SoundMixer.stopAll();
    }
    /***** TIMING AND CONTROL UNIT * *****/
    import flash.events.MouseEvent;

    stop();

    BtnTCL9.addEventListener(MouseEvent.CLICK, BtnTCLunit9)

    function BtnTCLunit9(event: MouseEvent): void {
        gotoAndStop(1, "timingAndControlUnit");
        SoundMixer.stopAll();
    }
    /***** REGISTERS *****/
    import flash.events.MouseEvent;

    stop();

    BtnReg9.addEventListener(MouseEvent.CLICK, BtnRegister9)

    function BtnRegister9(event: MouseEvent): void {
        gotoAndStop(1, "registers");
        SoundMixer.stopAll();
    }
    /***** DATA AND ADDRESS BUS *****/

```

```

import flash.events.MouseEvent;

stop();

BtnDataBus9.addEventListener(MouseEvent.CLICK, BtnDataAddressBus9)

function BtnDataAddressBus9(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
    SoundMixer.stopAll();
}

/***** TIMING AND CONTROL SIGNAL *****/
import flash.events.MouseEvent;

stop();

BtnTACS9.addEventListener(MouseEvent.CLICK, BtntimingAndCtrlSg9)

function BtntimingAndCtrlSg9(event: MouseEvent): void {
    gotoAndStop(1, "timmingAndControlSignal");
    SoundMixer.stopAll();
}

/***** FETCH OPERATION *****/
import flash.events.MouseEvent;

stop();

BtnFetch9.addEventListener(MouseEvent.CLICK, fetchOptBtn9)

function fetchOptBtn9(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}

/***** EXECUTE OPERATION *****/
import flash.events.MouseEvent;

stop();

BtnExec9.addEventListener(MouseEvent.CLICK, BtnExecute9)

function BtnExecute9(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}

/***** MACHINE CYCLE *****/
import flash.events.MouseEvent;

```

```
stop();
```

```
BtnMchnCycl9.addEventListener(MouseEvent.CLICK, BtnMachineCycle9)
```

```
function BtnMachineCycle9(event: MouseEvent): void {  
    gotoAndStop(1, "machineCycle");  
    SoundMixer.stopAll();  
}
```

```
/****** APPLICATIONS *****/  
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnApp9.addEventListener(MouseEvent.CLICK, BtnApplication9)
```

```
function BtnApplication9(event: MouseEvent): void {  
    gotoAndStop(1, "applications");  
    SoundMixer.stopAll();  
}
```

```
/****** 8085 INSTRUCTION *****/  
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnInstruc9.addEventListener(MouseEvent.CLICK, BtnInstruction9)
```

```
function BtnInstruction9(event: MouseEvent): void {  
    gotoAndStop(1, "instruction8085");  
    SoundMixer.stopAll();  
}
```

```
/******
```

```
ExecuteSound.addEventListener(MouseEvent.CLICK, sound11);
```

```
var fl_SC11:SoundChannel;
```

```
//This variable keeps track of whether you want to play or stop the sound
```

```
var fl_ToPlay11:Boolean = true;
```

```
function sound11(evt:MouseEvent):void
```

```
{  
    if(fl_ToPlay11)  
    {  
        var s11:Sound = new executeOperation;  
        fl_SC11 = s11.play();  
    }  
    else  
    {
```

```

        fl_SC11.stop();
    }
    fl_ToPlay11 = !fl_ToPlay11;
}
/*****
/***** Home Button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Home85.addEventListener(MouseEvent.CLICK, home3)
```

```

function home3(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
}
/***** Next button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
nextExecBtn.addEventListener(MouseEvent.CLICK, nextExec)
```

```

function nextExec(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
}
/***** Back button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
prevExecBtn.addEventListener(MouseEvent.CLICK, prevExec)
```

```

function prevExec(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
}
/*****

```

X-----Machine Cycle-----X

```

/***** WHOLE PAGE BUTTON NAVIGATION
*****/

```

```
/***** INTRODUCTION *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```



```
BtnIntro10.addEventListener(MouseEvent.CLICK, BtnIntro10)
```

```
function BtnIntro10(event: MouseEvent): void {  
    gotoAndStop(1, "introduction");  
    SoundMixer.stopAll();  
}  
/***** PIN DIAGRAM *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnPinDiagram10.addEventListener(MouseEvent.CLICK, BtnPinDiagrams10)
```

```
function BtnPinDiagrams10(event: MouseEvent): void {  
    gotoAndStop(1, "pinDiagram");  
    SoundMixer.stopAll();  
}  
/***** ARITHMETIC AND LOGIC UNIT *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnALU10.addEventListener(MouseEvent.CLICK, BtnALUnit10)
```

```
function BtnALUnit10(event: MouseEvent): void {  
    gotoAndStop(1, "alu");  
    SoundMixer.stopAll();  
}  
/***** TIMING AND CONTROL UNIT *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTCL10.addEventListener(MouseEvent.CLICK, BtnTCLUnit10)
```

```
function BtnTCLUnit10(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}  
/***** REGISTERS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnReg10.addEventListener(MouseEvent.CLICK, BtnRegister10)
```

```

function BtnRegister10(event: MouseEvent): void {
    gotoAndStop(1, "registers");
    SoundMixer.stopAll();
}
/***** DATA AND ADDRESS BUS *****/
import flash.events.MouseEvent;

stop();

BtnDataBus10.addEventListener(MouseEvent.CLICK, BtnDataAddressBus10)

function BtnDataAddressBus10(event: MouseEvent): void {
    gotoAndStop(1, "dataAndAddressBus");
    SoundMixer.stopAll();
}

/***** TIMING AND CONTROL SIGNAL *****/
import flash.events.MouseEvent;

stop();

BtnTACS10.addEventListener(MouseEvent.CLICK, BtntimingAndCtrlSg10)

function BtntimingAndCtrlSg10(event: MouseEvent): void {
    gotoAndStop(1, "timmingAndControlSignal");
    SoundMixer.stopAll();
}
/***** FETCH OPERATION *****/
import flash.events.MouseEvent;

stop();

BtnFetch10.addEventListener(MouseEvent.CLICK, fetchOptBtn10)

function fetchOptBtn10(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}
/***** EXECUTE OPERATION *****/
import flash.events.MouseEvent;

stop();

BtnExec10.addEventListener(MouseEvent.CLICK, BtnExecute10)

function BtnExecute10(event: MouseEvent): void {

```

```

        gotoAndStop(1, "executeOperation");
        SoundMixer.stopAll();
    }
/***** MACHINE CYCLE *****/
import flash.events.MouseEvent;

stop();

BtnMchnCycl10.addEventListener(MouseEvent.CLICK, BtnMachineCycle10)

function BtnMachineCycle10(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
/***** APPLICATIONS *****/
import flash.events.MouseEvent;

stop();

BtnApp10.addEventListener(MouseEvent.CLICK, BtnApplication10)

function BtnApplication10(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

BtnInstruc10.addEventListener(MouseEvent.CLICK, BtnInstruction10)

function BtnInstruction10(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}
/*****
MachineCycleSound.addEventListener(MouseEvent.CLICK, sound12);

var fl_SC12:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay12:Boolean = true;

function sound12(evt:MouseEvent):void
{
    if(fl_ToPlay12)
    {

```

```

                var s12:Sound = new machineCycle;
                fl_SC12 = s12.play();
            }
            else
            {
                fl_SC12.stop();
            }
            fl_ToPlay12 = !fl_ToPlay12;
        }
/*****
/***** Home Button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Home90.addEventListener(MouseEvent.CLICK, home2)
```

```
function home2(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
/***** Next button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
nextMchnBtn.addEventListener(MouseEvent.CLICK, nextMchn)
```

```
function nextMchn(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** Back button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
prevMchnBtn.addEventListener(MouseEvent.CLICK, prevMchn)
```

```
function prevMchn(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}
/*****

```

X-----Applications-----X

```

/***** next Inner Btn *****/
stop();

innerNextBtn.addEventListener(MouseEvent.CLICK, nextInner);

function nextInner(event:MouseEvent):void
{
    nextFrame();
}

/***** Home Button *****/

import flash.events.MouseEvent;

stop();

Home95.addEventListener(MouseEvent.CLICK, home)

function home(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}
/***** Next button *****/

import flash.events.MouseEvent;

stop();

nextAppBtn.addEventListener(MouseEvent.CLICK, nextapp)

function nextapp(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
    SoundMixer.stopAll();
}

/***** Back button *****/

import flash.events.MouseEvent;

stop();

prevAppBtn.addEventListener(MouseEvent.CLICK, prevapp)

function prevapp(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
}
```

```

        SoundMixer.stopAll();
    }
    /*****

    *****/

    /***** Back Inner btn *****/

    stop();

    backInnerBtn.addEventListener(MouseEvent.CLICK, prevInner);

    function prevInner(event:MouseEvent):void
    {
        prevFrame();
    }

    /***** Home Button *****/

    import flash.events.MouseEvent;

    stop();

    Home95.addEventListener(MouseEvent.CLICK, home950)

    function home950(event: MouseEvent): void {
        gotoAndStop(1, "ContentPage");
        SoundMixer.stopAll();
    }
    /***** Next button *****/

    import flash.events.MouseEvent;

    stop();

    nextAppBtn.addEventListener(MouseEvent.CLICK, nextapp950)

    function nextapp950(event: MouseEvent): void {
        gotoAndStop(1, "instruction8085");
        SoundMixer.stopAll();
    }

    /***** Back button *****/

    import flash.events.MouseEvent;

    stop();

    prevAppBtn.addEventListener(MouseEvent.CLICK, prevapp950)

```

```

function prevapp950(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
/*****

/***** WHOLE PAGE BUTTON NAVIGATION
*****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro11.addEventListener(MouseEvent.CLICK, BtnIntro11)

function BtnIntro11(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram11.addEventListener(MouseEvent.CLICK, BtnPinDiagrams11)

function BtnPinDiagrams11(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/***** ARITHMATIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU11.addEventListener(MouseEvent.CLICK, BtnALUnit11)

function BtnALUnit11(event: MouseEvent): void {
    gotoAndStop(1, "alu");
    SoundMixer.stopAll();
}
/***** TIMING AND CONTROL UNIT * *****/
import flash.events.MouseEvent;

stop();

```

```
BtnTCL11.addEventListener(MouseEvent.CLICK, BtnTCLUnit11)
```

```
function BtnTCLUnit11(event: MouseEvent): void {  
    gotoAndStop(1, "timingAndControlUnit");  
    SoundMixer.stopAll();  
}  
/***** REGISTERS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnReg11.addEventListener(MouseEvent.CLICK, BtnRegister11)
```

```
function BtnRegister11(event: MouseEvent): void {  
    gotoAndStop(1, "registers");  
    SoundMixer.stopAll();  
}  
/***** DATA AND ADDRESS BUS *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnDataBus11.addEventListener(MouseEvent.CLICK, BtnDataAddressBus11)
```

```
function BtnDataAddressBus11(event: MouseEvent): void {  
    gotoAndStop(1, "dataAndAddressBus");  
    SoundMixer.stopAll();  
}  
  
/***** TIMING AND CONTROL SIGNAL *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnTACS11.addEventListener(MouseEvent.CLICK, BntimingAndCtrlSg11)
```

```
function BntimingAndCtrlSg11(event: MouseEvent): void {  
    gotoAndStop(1, "timmingAndControlSignal");  
    SoundMixer.stopAll();  
}  
/***** FETCH OPERATION *****/  
import flash.events.MouseEvent;  
  
stop();
```

```
BtnFetch11.addEventListener(MouseEvent.CLICK, fetchOptBtn11)
```



```

function fetchOptBtn11(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}
/***** EXECUTE OPERATION *****/
import flash.events.MouseEvent;

stop();

```

```

BtnExec11.addEventListener(MouseEvent.CLICK, BtnExecute11)

```

```

function BtnExecute11(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}
/***** MACHINE CYCLE *****/
import flash.events.MouseEvent;

stop();

```

```

BtnMchnCycl11.addEventListener(MouseEvent.CLICK, BtnMachineCycle11)

```

```

function BtnMachineCycle11(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
/***** APPLICATIONS *****/
import flash.events.MouseEvent;

stop();

```

```

BtnApp11.addEventListener(MouseEvent.CLICK, BtnApplication11)

```

```

function BtnApplication11(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/***** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;

stop();

```

```

BtnInstruc11.addEventListener(MouseEvent.CLICK, BtnInstruction11)

```

```

function BtnInstruction11(event: MouseEvent): void {
    gotoAndStop(1, "instruction8085");
}

```

```

        SoundMixer.stopAll();
    }
    /*****
ApplicationSound.addEventListener(MouseEvent.CLICK, sound13);

var fl_SC13:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay13:Boolean = true;

function sound13(evt:MouseEvent):void
{
    if(fl_ToPlay13)
    {
        var s13:Sound = new application;
        fl_SC13 = s13.play();
    }
    else
    {
        fl_SC13.stop();
    }
    fl_ToPlay13 = !fl_ToPlay13;
}
*****/

```

X-----Instruction 8085-----X

```

/*****/

dtgrp.addEventListener(MouseEvent.CLICK, fl_ClickToGoToAndStopAtFrame_4);

function fl_ClickToGoToAndStopAtFrame_4(event:MouseEvent):void
{
    gotoAndStop(10);
}

/*****/

arthgrp.addEventListener(MouseEvent.CLICK, f2_ClickToGoToAndStopAtFrame_4);

function f2_ClickToGoToAndStopAtFrame_4(event:MouseEvent):void
{
    gotoAndStop(22);
}

/*****/

loggrp.addEventListener(MouseEvent.CLICK, f3_ClickToGoToAndStopAtFrame_4);

function f3_ClickToGoToAndStopAtFrame_4(event:MouseEvent):void
{

```

```

        gotoAndStop(32);
    }

    /*****

branchgrp.addEventListener(MouseEvent.CLICK, f4_ClickToGoToAndStopAtFrame_4);

function f4_ClickToGoToAndStopAtFrame_4(event:MouseEvent):void
{
    gotoAndStop(43);
}

    *****/

stackgrp.addEventListener(MouseEvent.CLICK, f5_ClickToGoToAndStopAtFrame_4);

function f5_ClickToGoToAndStopAtFrame_4(event:MouseEvent):void
{
    gotoAndStop(48);
}

    *****/

    /***** Home Button *****/

import flash.events.MouseEvent;

stop();

Home100.addEventListener(MouseEvent.CLICK, home100)

function home100(event: MouseEvent): void {
    gotoAndStop(1, "ContentPage");
    SoundMixer.stopAll();
}

    *****/ Next button *****/

import flash.events.MouseEvent;

stop();

nextInstrBtn.addEventListener(MouseEvent.CLICK, nextInstr100)

function nextInstr100(event: MouseEvent): void {
    gotoAndStop(1, "simulator");
    SoundMixer.stopAll();
}

    *****/ Back button *****/

```

```

import flash.events.MouseEvent;

stop();

prevInstrBtn.addEventListener(MouseEvent.CLICK, prevInstr100)

function prevInstr100(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
/*****
/***** WHOLE PAGE BUTTON NAVIGATION
*****/

/***** INTRODUCTION *****/
import flash.events.MouseEvent;

stop();

BtnIntro12.addEventListener(MouseEvent.CLICK, BtnIntro12)

function BtnIntro12(event: MouseEvent): void {
    gotoAndStop(1, "introduction");
    SoundMixer.stopAll();
}
/***** PIN DIAGRAM *****/
import flash.events.MouseEvent;

stop();

BtnPinDiagram12.addEventListener(MouseEvent.CLICK, BtnPinDiagrams12)

function BtnPinDiagrams12(event: MouseEvent): void {
    gotoAndStop(1, "pinDiagram");
    SoundMixer.stopAll();
}
/***** ARITHMETIC AND LOGIC UNIT *****/
import flash.events.MouseEvent;

stop();

BtnALU12.addEventListener(MouseEvent.CLICK, BtnALUnit12)

function BtnALUnit12(event: MouseEvent): void {
    gotoAndStop(1, "alu");

```

```

        SoundMixer.stopAll();
    }
    /***** TIMING AND CONTROL UNIT *****/
    import flash.events.MouseEvent;

    stop();

    BtnTCL12.addEventListener(MouseEvent.CLICK, BtnTCLunit12)

    function BtnTCLunit12(event: MouseEvent): void {
        gotoAndStop(1, "timingAndControlUnit");
        SoundMixer.stopAll();
    }
    /***** REGISTERS *****/
    import flash.events.MouseEvent;

    stop();

    BtnReg12.addEventListener(MouseEvent.CLICK, BtnRegister12)

    function BtnRegister12(event: MouseEvent): void {
        gotoAndStop(1, "registers");
        SoundMixer.stopAll();
    }
    /***** DATA AND ADDRESS BUS *****/
    import flash.events.MouseEvent;

    stop();

    BtnDataBus12.addEventListener(MouseEvent.CLICK, BtnDataAddressBus12)

    function BtnDataAddressBus12(event: MouseEvent): void {
        gotoAndStop(1, "dataAndAddressBus");
        SoundMixer.stopAll();
    }

    /***** TIMING AND CONTROL SIGNAL *****/
    import flash.events.MouseEvent;

    stop();

    BtnTACS12.addEventListener(MouseEvent.CLICK, BtntimingAndCtrlSg12)

    function BtntimingAndCtrlSg12(event: MouseEvent): void {
        gotoAndStop(1, "timmingAndControlSignal");
        SoundMixer.stopAll();
    }

```

```
/****** FETCH OPERATION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnFetch12.addEventListener(MouseEvent.CLICK, fetchOptBtn12)
```

```
function fetchOptBtn12(event: MouseEvent): void {
    gotoAndStop(1, "fetchOperation");
    SoundMixer.stopAll();
}
```

```
/****** EXECUTE OPERATION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnExec12.addEventListener(MouseEvent.CLICK, BtnExecute12)
```

```
function BtnExecute12(event: MouseEvent): void {
    gotoAndStop(1, "executeOperation");
    SoundMixer.stopAll();
}
```

```
/****** MACHINE CYCLE *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnMchnCycl12.addEventListener(MouseEvent.CLICK, BtnMachineCycle12)
```

```
function BtnMachineCycle12(event: MouseEvent): void {
    gotoAndStop(1, "machineCycle");
    SoundMixer.stopAll();
}
```

```
/****** APPLICATIONS *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnApp12.addEventListener(MouseEvent.CLICK, BtnApplication12)
```

```
function BtnApplication12(event: MouseEvent): void {
    gotoAndStop(1, "applications");
    SoundMixer.stopAll();
}
```

```
/****** 8085 INSTRUCTION *****/
import flash.events.MouseEvent;
```

```
stop();
```

```
BtnInstruc12.addEventListener(MouseEvent.CLICK, BtnInstruction12)
```

```
function BtnInstruction12(event: MouseEvent): void {  
    gotoAndStop(1, "instruction8085");  
    SoundMixer.stopAll();  
}
```

```
/****** Coding Btn *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
CodingBtn.addEventListener(MouseEvent.CLICK, coding)
```

```
function coding(event: MouseEvent): void {  
    gotoAndStop(1, "coding 1");  
    SoundMixer.stopAll();  
}
```

```
/****** Quiz btn *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
QuizSection.addEventListener(MouseEvent.CLICK, Quiz)
```

```
function Quiz(event: MouseEvent): void {  
    gotoAndStop(1, "quiz 1");  
    SoundMixer.stopAll();  
}
```

```
/****** Simulator Btn *****/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
simulatorBtn1.addEventListener(MouseEvent.CLICK, simulatorBtn)
```

```
function simulatorBtn(event: MouseEvent): void {  
    gotoAndStop(1, "simulator");  
    SoundMixer.stopAll();  
}
```

```
/******
```

```
InstructionSound.addEventListener(MouseEvent.CLICK, sound14);
```

```
var fl_SC14:SoundChannel;
```

```
//This variable keeps track of whether you want to play or stop the sound
```

```
var fl_ToPlay14:Boolean = true;
```

```
function sound14(evt:MouseEvent):void
```

```
{
    if(fl_ToPlay14)
    {
        var s14:Sound = new instruction8085;
        fl_SC14 = s14.play();
    }
    else
    {
        fl_SC14.stop();
    }
    fl_ToPlay14 = !fl_ToPlay14;
}
```

```
/*-----X-----Simulator-----X-----*/
```

```
step1Coding.addEventListener(MouseEvent.CLICK, fxy123456);
```

```
function fxy123456(event:MouseEvent):void
```

```
{
    gotoAndStop(20);
    SoundMixer.stopAll();
}
```

```
/*-----X-----Simulator-----X-----*/
```

```
SimulatorSound1.addEventListener(MouseEvent.CLICK, sound15);
```

```
var fl_SC15:SoundChannel;
```

```
//This variable keeps track of whether you want to play or stop the sound
```

```
var fl_ToPlay15:Boolean = true;
```

```
function sound15(evt:MouseEvent):void
```

```
{
    if(fl_ToPlay15)
    {
        var s15:Sound = new simulator;
        fl_SC15 = s15.play();
    }
    else
    {
        fl_SC15.stop();
    }
    fl_ToPlay15 = !fl_ToPlay15;
}
```



```

/*****/
/*****/
codingSec2Btn..addEventListener(MouseEvent.CLICK, fxy1000);

function fxy1000(event:MouseEvent):void
{
    gotoAndStop(20);
    SoundMixer.stopAll();
}
/*****/
/*****/
step2Btn.addEventListener(MouseEvent.CLICK, step2Btn10);

function step2Btn10(event: MouseEvent): void {
    gotoAndStop(195);
}

/*****/
nextBtnSim.addEventListener(MouseEvent.CLICK, fxy305);

function fxy305(event:MouseEvent):void
{
    gotoAndStop(305);
}
/*****/
/***** Next button *****/

import flash.events.MouseEvent;

stop();

nextSimBtn10.addEventListener(MouseEvent.CLICK, nextSim)

function nextSim(event: MouseEvent): void {
    gotoAndStop(1, "coding 1");
    SoundMixer.stopAll();
}
/*****/

step1Coding.addEventListener(MouseEvent.CLICK, fxy789);

function fxy789(event:MouseEvent):void
{
    gotoAndStop(10);
}
/*****/
/*reset to start*/
stop();

reset.addEventListener(MouseEvent.CLICK, resetBtn5123);

```

```

function resetBtn5123(event: MouseEvent): void {
    gotoAndStop(25);
}
/*****/

/*Exam memory*/
stop();

relExmem.addEventListener(MouseEvent.CLICK, examMemory);

function examMemory(event: MouseEvent): void {
    gotoAndStop(30);
}

/*****/
/***** 2000H *****/
/***** 2 *****/
ser2Btn.addEventListener(MouseEvent.CLICK, ser2);

function ser2(event: MouseEvent): void {
    gotoAndStop(35);
}
/*****/
Frame 30 to frame 360 same code with different string name ,instance name and refered frame
number
and in code 360 code as follows
/*****/
nextSimBtn10.addEventListener(MouseEvent.CLICK,nextC1123)

function nextC1123(event: MouseEvent): void {
    gotoAndStop(1, "coding 1");
    SoundMixer.stopAll();
}

/*****/

```

X-----Coding 1-----X

```

/***** Coding 1 *****/

import flash.events.MouseEvent;

stop();

submitBtn1.addEventListener(MouseEvent.CLICK, checkClick1);

function checkClick1(event: MouseEvent): void {

    if (codingtext1.text == "LHLD 4000H/XCHG/LHLD 4002H/MOV A,E/ADD L/MOV L,A/MOV
A,D/ADC H/MOV H,A/SHLD 4004H/HLT") {
        ansCheck.text = "Great ! You did a great job. Please press next to proceed.";
    }
}

```

```

        } else {
            ansCheck.text = "Incorrect! The correct answer is: LHL 4000H/XCHG/LHL 4002H/MOV A, E/ADD L/MOV L, A/MOV A, D/ADC H/MOV H, A/SHL 4004H/HLT. Please press next button to go 2nd Coding";
        }
    }
}

```

```

/*****

```

```

/***** Next button *****/

```

```

import flash.events.MouseEvent;

stop();

nextBtnCode1.addEventListener(MouseEvent.CLICK, nextC1)

function nextC1(event: MouseEvent): void {
    gotoAndStop(1, "coding 2");
    SoundMixer.stopAll();
}

```

```

/***** Prev button *****/

```

```

import flash.events.MouseEvent;

stop();

prevBtnCode1.addEventListener(MouseEvent.CLICK, prevC1)

function prevC1(event: MouseEvent): void {
    gotoAndStop(1, "simulator");
    SoundMixer.stopAll();
}

```

```

/*****

```

X-----Coding 2-----X

```

import flash.events.MouseEvent;

stop();

submitBtn2.addEventListener(MouseEvent.CLICK, checkClick2);

function checkClick2(event: MouseEvent): void
{

```

```

if (codingtext2.text == "LXI H 4000H/MOV A,M/INX H /SUB M/INX H/MOV M,A/HLT")
{
    ansCheck2.text = "Great ! You did a great job. Please press next to proceed.";
}
else {codingtext2.text == "";
    ansCheck2.text = "Incorrect! The correct answer is: LXI H 4000H/MOV A,M/INX H /SUB
M/INX H/MOV M,A/HLT. Please press next button to go 3rd Coding";
}}

```

```

/*****

```

```

/***** Next button *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

nextBtnCode2.addEventListener(MouseEvent.CLICK,nextC2)

```

```

function nextC2(event: MouseEvent): void {
    gotoAndStop(1, "coding 3");
    SoundMixer.stopAll();
}

```

```

/***** Prev button *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

prevBtnCode2.addEventListener(MouseEvent.CLICK,prevC2)

```

```

function prevC2(event: MouseEvent): void {
    gotoAndStop(1, "coding 1");
    SoundMixer.stopAll();
}

```

```

/*****

```

X-----Coding 3-----X

```

/***** code3 *****/

```

```

import flash.events.MouseEvent;

```

```

stop();

```

```

submitBtn3.addEventListener(MouseEvent.CLICK, checkClick3);

```

```

function checkClick3(event: MouseEvent): void

```

```

{
if (codingtext3.text == "LXI H 2000H/LXI D 4000H/MOV B, M/LDAX D/MOV M, A/MOV A, B/STAX
D/HLT")
{
    ansCheck3.text = "Great ! You did a great job. Please press next to proceed.";
}
else {codingtext3.text == "";
    ansCheck3.text = "Incorrect ! The correct answer is: LXI H 2000H/LXI D 4000H/MOV B,
M/LDAX D/MOV M, A/MOV A, B/STAX D/HLT. Please press next to start quiz section";
}}

```

```

/*****/
/***** Next button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
nextBtnCode3.addEventListener(MouseEvent.CLICK,nextC3)
```

```

function nextC3(event: MouseEvent): void {
    gotoAndStop(1, "quiz 1");
    SoundMixer.stopAll();
}

```

```

/*****/
/***** Prev button *****/

```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
prevBtnCode3.addEventListener(MouseEvent.CLICK,prevC3)
```

```

function prevC3(event: MouseEvent): void {
    gotoAndStop(1, "coding 2");
    SoundMixer.stopAll();
}

```

```

/*****/

```

X-----Quiz 1-----X

```
stop();
```

```

/*****/

```

```
optA.addEventListener(MouseEvent.CLICK, fl_Frame_4);
```

```
function fl_Frame_4(event:MouseEvent):void
```

```

{
    gotoAndStop(32);
}

/*****

optB.addEventListener(MouseEvent.CLICK, f2_Frame_4);

function f2_Frame_4(event:MouseEvent):void
{
    gotoAndStop(47);
}

/*****

optC.addEventListener(MouseEvent.CLICK, f3_Frame_4);

function f3_Frame_4(event:MouseEvent):void
{
    gotoAndStop(16);
}

/*****

optD.addEventListener(MouseEvent.CLICK, f4_Frame_4);

function f4_Frame_4(event:MouseEvent):void
{
    gotoAndStop(61);
}

/*****
/***** Next button *****/

import flash.events.MouseEvent;

stop();

Q1nextBtn.addEventListener(MouseEvent.CLICK, nextQ1)

function nextQ1(event: MouseEvent): void {
    gotoAndStop(1, "quiz 2");
    SoundMixer.stopAll();
}

/***** Prev Btn *****/

import flash.events.MouseEvent;

stop();

```

```
Q1prevBtn.addEventListener(MouseEvent.CLICK, prevQ1)
```

```
function prevQ1(event: MouseEvent): void {  
    gotoAndStop(1, "coding 3");  
    SoundMixer.stopAll();  
}
```

```
/******
```

X-----Quiz 2-----X

```
/* Click to Go to Frame and Stop
```

Clicking on the specified symbol instance moves the playhead to the specified frame in the timeline and stops the movie.

Can be used on the main timeline or on movie clip timelines.

Instructions:

1. Replace the number 5 in the code below with the frame number you would like the playhead to move to when the symbol instance is clicked.

```
*/
```

```
stop();
```

```
/******
```

```
opt2A.addEventListener(MouseEvent.CLICK, fl_Frame);
```

```
function fl_Frame(event: MouseEvent): void {  
    gotoAndStop(16);  
}
```

```
/******
```

```
opt2B.addEventListener(MouseEvent.CLICK, f2_Frame);
```

```
function f2_Frame(event: MouseEvent): void {  
    gotoAndStop(31);  
}
```

```
/******
```

```
opt2C.addEventListener(MouseEvent.CLICK, f3_Frame);
```

```
function f3_Frame(event: MouseEvent): void {  
    gotoAndStop(46);  
}
```

```
/******
```

```
opt2D.addEventListener(MouseEvent.CLICK, f4_Frame);
```

```
function f4_Frame(event: MouseEvent): void {
    gotoAndStop(61);
}
```

```
/******  
***** Next button ******/
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Q2nextBtn.addEventListener(MouseEvent.CLICK,nextQ2)
```

```
function nextQ2(event: MouseEvent): void {
    gotoAndStop(1, "quiz 3");
    SoundMixer.stopAll();
}
```

```
/******
```

X-----Quiz 3-----X

```
stop();
```

```
/******
```

```
opt3A.addEventListener(MouseEvent.CLICK, Optxa);
```

```
function Optxa(event: MouseEvent): void {
    gotoAndStop(23);
}
```

```
/******
```

```
opt3B.addEventListener(MouseEvent.CLICK, Optxb);
```

```
function Optxb(event: MouseEvent): void {
    gotoAndStop(46);
}
```

```
/******
```

```
opt3C.addEventListener(MouseEvent.CLICK, Optxc);
```

```
function Optxc(event: MouseEvent): void {
    gotoAndStop(62);
}
```

```
/******
```

```
opt3D.addEventListener(MouseEvent.CLICK, Optxd);
```



```
function Optxd(event: MouseEvent): void {
    gotoAndStop(81);
}
```

```
/***/
```

```
/***/ Next button */
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Q3nextBtn.addEventListener(MouseEvent.CLICK, nextQ3)
```

```
function nextQ3(event: MouseEvent): void {
    gotoAndStop(1, "matching");
    SoundMixer.stopAll();
}
```

```
/***/
```

```
/***/ Next button */
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Q3nextBtn.addEventListener(MouseEvent.CLICK, nextQ312)
```

```
function nextQ312(event: MouseEvent): void {
    gotoAndStop(1, "matching");
    SoundMixer.stopAll();
}
```

```
/***/
```

```
/***/ Next button */
```

```
import flash.events.MouseEvent;
```

```
stop();
```

```
Q3nextBtn.addEventListener(MouseEvent.CLICK, nextQ31)
```

```
function nextQ31(event: MouseEvent): void {
    gotoAndStop(1, "matching");
    SoundMixer.stopAll();
}
```

```
/***/
```

X-----Matching-----X

```

/***** matching *****/
branchGroup.addEventListener(MouseEvent.CLICK, drag);
stage.addEventListener(MouseEvent.CLICK, drop);
/*-----*/
memoryMappedIO.addEventListener(MouseEvent.CLICK, drag);
stage.addEventListener(MouseEvent.CLICK, drop);
/*-----*/
dataTransferGroup.addEventListener(MouseEvent.CLICK, drag);
stage.addEventListener(MouseEvent.CLICK, drop);
/*-----*/
peripheralMappedIO.addEventListener(MouseEvent.CLICK, drag);
stage.addEventListener(MouseEvent.CLICK, drop);
/*-----*/
nonMaskableInterrupt.addEventListener(MouseEvent.CLICK, drag);
stage.addEventListener(MouseEvent.CLICK, drop);
/*-----*/

function drag(e: MouseEvent): void {
    e.target.startDrag();
    btnCheck200.text = "Keep trying";
}

/*-----*/
function drop(e: MouseEvent): void {
    stopDrag();
    if (branchGroup.hitTestObject(jmpBtn)) {
        if (memoryMappedIO.hitTestObject(bit16dvcAdds)) {
            if (dataTransferGroup.hitTestObject(movBtn)) {
                if (peripheralMappedIO.hitTestObject(bit8dvcAdds)) {
                    if (nonMaskableInterrupt.hitTestObject(trapBtn)) {
                        /*submitBtnMatch*/
                        btnCheck200.text = "Well done ! You got the correct
answer.";
                    }
                }
            }
        }
    }
} else {
    btnCheck200.text = "Incorrect! Reset and try again to proceed.";
}

}

/*-----*/
/*****Reset Button *****/

resetBtn.addEventListener(MouseEvent.CLICK, posClick);
function posClick(event: MouseEvent): void {
    btnCheck200.text = " ";
    memoryMappedIO.x = 478.4;
    memoryMappedIO.y = 201.2;
}

```

```

        nonMaskableInterrupt.x = 478.4;
        nonMaskableInterrupt.y = 272.5;

        peripheralMappedIO.x = 478.4;
        peripheralMappedIO.y = 340.9;

        dataTransferGroup.x = 478.4;
        dataTransferGroup.y = 411.35;

        branchGroup.x = 478.4;
        branchGroup.y = 479.6;

    }
    /***** Next button *****/

import flash.events.MouseEvent;

stop();

nextMatchBtn.addEventListener(MouseEvent.CLICK, nextMatch10)

function nextMatch10(event: MouseEvent): void {
    gotoAndStop(1, "sequence");
    SoundMixer.stopAll();
}
/*****/
/*****/
matchingBtnSound.addEventListener(MouseEvent.CLICK, sound20);

var fl_SC20:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay20:Boolean = true;

function sound20(evt:MouseEvent):void
{
    if(fl_ToPlay20)
    {
        var s20:Sound = new matchTheFollowingSound;
        fl_SC20 = s20.play();
    }
    else
    {
        fl_SC20.stop();
    }
    fl_ToPlay20 = !fl_ToPlay20;
}
/*****/

```

~~X-----Sequence-----X~~

```
stop();
/*****1***** halt combo *****/
HltCombo76.addItem({
    label: "Choose the correct option"
});
HltCombo76.addItem({
    label: "73"
});
HltCombo76.addItem({
    label: "76"
});
HltCombo76.addItem({
    label: "74"
});
/*****text design*****/
var tfor: TextFormat = new TextFormat();

tfor.size = 25;

HltCombo76.textField.setStyle("textFormat", tfor);

HltCombo76.setStyle("disabledTextFormat", tfor);

HltCombo76.dropdown.setRendererStyle("Trebuchet MS", tfor);
/*****/

HltCombo76.addEventListener(Event.CHANGE, changeOption1);

function changeOption1(event: Event): void {
    if (HltCombo76.selectedItem.label == "73") gotoAndStop(5);
    if (HltCombo76.selectedItem.label == "76") gotoAndStop(10);
    if (HltCombo76.selectedItem.label == "74") gotoAndStop(15);
}

/*****2***** exchange combo *****/
xchgComboEB.addItem({
    label: "Choose the correct option"
});
xchgComboEB.addItem({
    label: "EB"
});
xchgComboEB.addItem({
    label: "EC"
});
xchgComboEB.addItem({
    label: "EF"
});
/*****text design*****/
var tfor1: TextFormat = new TextFormat();
```

```

tfor1.size = 25;

xchgComboEB.textField.setStyle("textFormat", tfor1);

xchgComboEB.setStyle("disabledTextFormat", tfor1);

xchgComboEB.dropdown.setRendererStyle("Trebuchet MS", tfor1);
/*****/

xchgComboEB.addEventListener(Event.CHANGE, changeOption2);

function changeOption2(event: Event): void {
    if (xchgComboEB.selectedItem.label == "EB") gotoAndStop(20);
    if (xchgComboEB.selectedItem.label == "EC") gotoAndStop(25);
    if (xchgComboEB.selectedItem.label == "EF") gotoAndStop(30);
}
/*****3***** lxi h combo *****/
lxihCombo21.addItem({
    label: "Choose the correct option"
});
lxihCombo21.addItem({
    label: "21"
});
lxihCombo21.addItem({
    label: "24"
});
lxihCombo21.addItem({
    label: "23"
});

/*****text design*****/
var tfor2: TextFormat = new TextFormat();

tfor2.size = 25;

lxihCombo21.textField.setStyle("textFormat", tfor2);

lxihCombo21.setStyle("disabledTextFormat", tfor2);

lxihCombo21.dropdown.setRendererStyle("Trebuchet MS", tfor2);
/*****/

lxihCombo21.addEventListener(Event.CHANGE, changeOption3);

function changeOption3(event: Event): void {
    if (lxihCombo21.selectedItem.label == "21") gotoAndStop(35);
    if (lxihCombo21.selectedItem.label == "24") gotoAndStop(40);
    if (lxihCombo21.selectedItem.label == "23") gotoAndStop(45);
}

```

```

}

/*****/

checksequenceBtn.addEventListener(MouseEvent.CLICK, checksequence);

function checksequence(event:MouseEvent):void
{
    gotoAndStop(50);
}
/*****/
/***** Next button *****/

import flash.events.MouseEvent;
stop();
nextSqncBtn.addEventListener(MouseEvent.CLICK, endSceneBtn)

function endSceneBtn(event: MouseEvent): void {
    gotoAndStop(1, "endScene");
}
/*****/
nextSqncBtn.addEventListener(MouseEvent.CLICK, sound201);

var fl_SC201:SoundChannel;

//This variable keeps track of whether you want to play or stop the sound
var fl_ToPlay201:Boolean = true;

function sound201(evt:MouseEvent):void
{
    if(fl_ToPlay201)
    {
        var s201:Sound = new endPage;
        fl_SC201 = s201.play();
    }
    else
    {
        fl_SC201.stop();
    }
    fl_ToPlay201 = !fl_ToPlay201;
}
/*****/

X-----End Scene-----X

/*****/
exitFromCBT.addEventListener(MouseEvent.CLICK, fxName);
function fxName(event:MouseEvent):void
{
    fscommand("quit");
}
/*****/

```

X----- Audio files used -----X











Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\audio








Format: MP3 File (.mp3)

Bit rate: 320 kbps

Sample rate: 44100 Hz

Audio files names with individual details:

File Name	Voice over (button Instance name with reference)	Size (KB)	Length (Sec.)	Scene name (where will be played)	Main File
1. welcomePageCorrected.mp3	soundScene2 <BM2.1>	388.0	09	Welcome Page (Scene 2)	 welcomePageCorrected.mp3
2. contentAudio.mp3	SoundContent <BM3.1>	560.3	13	Content Page (Scene 3)	 contentAudio.mp3
3. introduction.mp3	IntroSound <BM4.1>	474.1	11	Introduction (Scene 4)	 introduction.mp3
4. pinDiagram.mp3	PinDiagramSound <BM5.1>	474.1	11	Pin Diagram (Scene 5)	 pinDiagram.mp3
5. alu.mp3	ALUSound <BM6.1>	603.4	14	ALU (Scene 6)	 alu.mp3
6. Timing and Control Unit.mp3	TCUSound <BM7.1>	388.0	09	Timing and Control unit (Scene 7)	 Timing and Control Unit.mp3
7. registers.mp3	RegSound <BM8.1>	344.8	08	Registers (Scene 8)	 registers.mp3
8. data and address bus.mp3	DataAddressBusSound <BM9.1>	258.6	06	Data and Address Bus (Scene 9)	 data and address bus.mp3
9. timing control signal.mp3	TimingAndControlSignal Sound <BM10.1>	431.1	10	Timing and Control Signal (Scene 10)	 timing control signal.mp3
10. fetchOperation.mp3	FetchSound <BM11.1>	905.1	21	Fetch Operation (Scene 11)	 fetchOperation.mp3

11. execute Operation.mp3	ExecuteSound <BM12.1>	517.2	12	Execute Operation (Scene12)	 executeOperation. mp3
12. machineCycle.mp3	MachineCycleSound <BM13.1>	431.1	10	Machine Cycle (Scene 13)	 machineCycle.mp3
13. applications.mp3	ApplicationSound <BM14.1>	1034.4	24	Applications (Scene 14)	 application.mp3
14. instruction8085.mp3	InstructionSound <BM15.1>	646.5	15	Instruction 8085 (Scene 15)	 instruction8085.mp 3
15. simulator.mp3	SimulatorSound1 <BM16.1>	819.1	19	Simulator (Scene 16)	 simulator.mp3
16. matchTheFollowing.mp3	MatchingBtnSound <BM23.1>	258.6	06	Matching (Scene 23)	 matchTheFollowing Sound.mp3
17. endPage.mp3	NextSqncBtn <BM24.1>	301.7	07	End page (Scene25)	 endPage.mp3

X----- Image used -----X

Used image as background at end page:

Size-113 KB (1, 16,641 bytes)

Format-JPG File (.jpg)

Location: C:\Users\Avirup Saha\Desktop\CBT IT CWE 2020-22\images

JPG Name- Integrated circuits and components

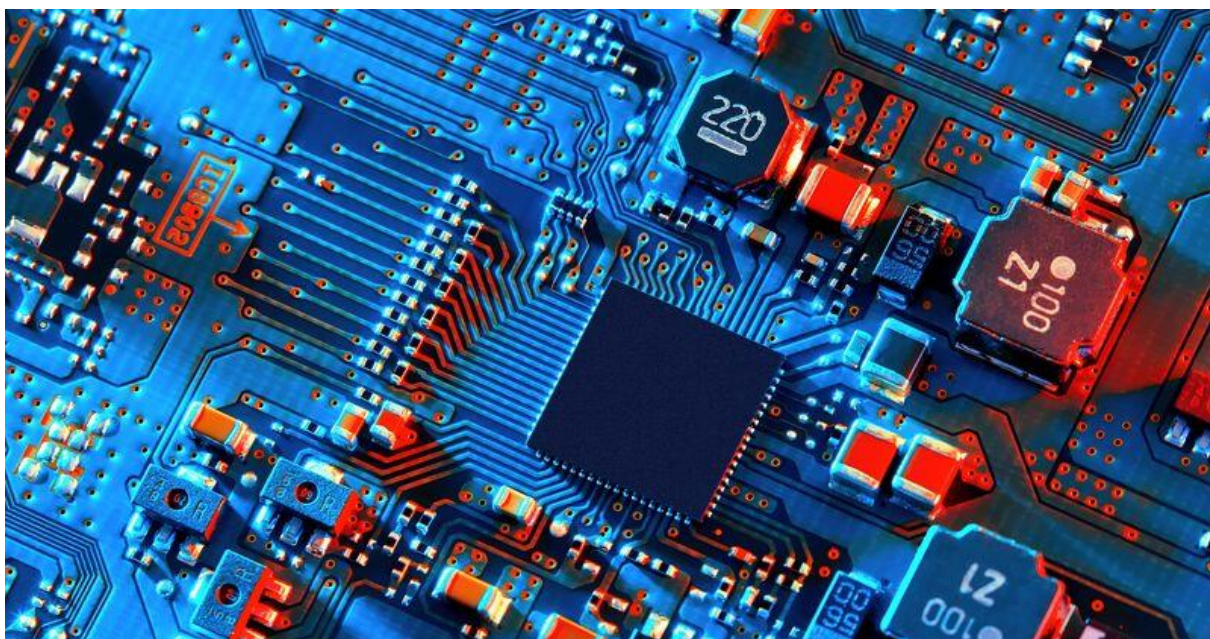
Height: 428 pixels

Width: 800 pixels

Dimensions: 800 X 428

Horizontal and Vertical resolution 96 dpi

Source: Internet



X-----X