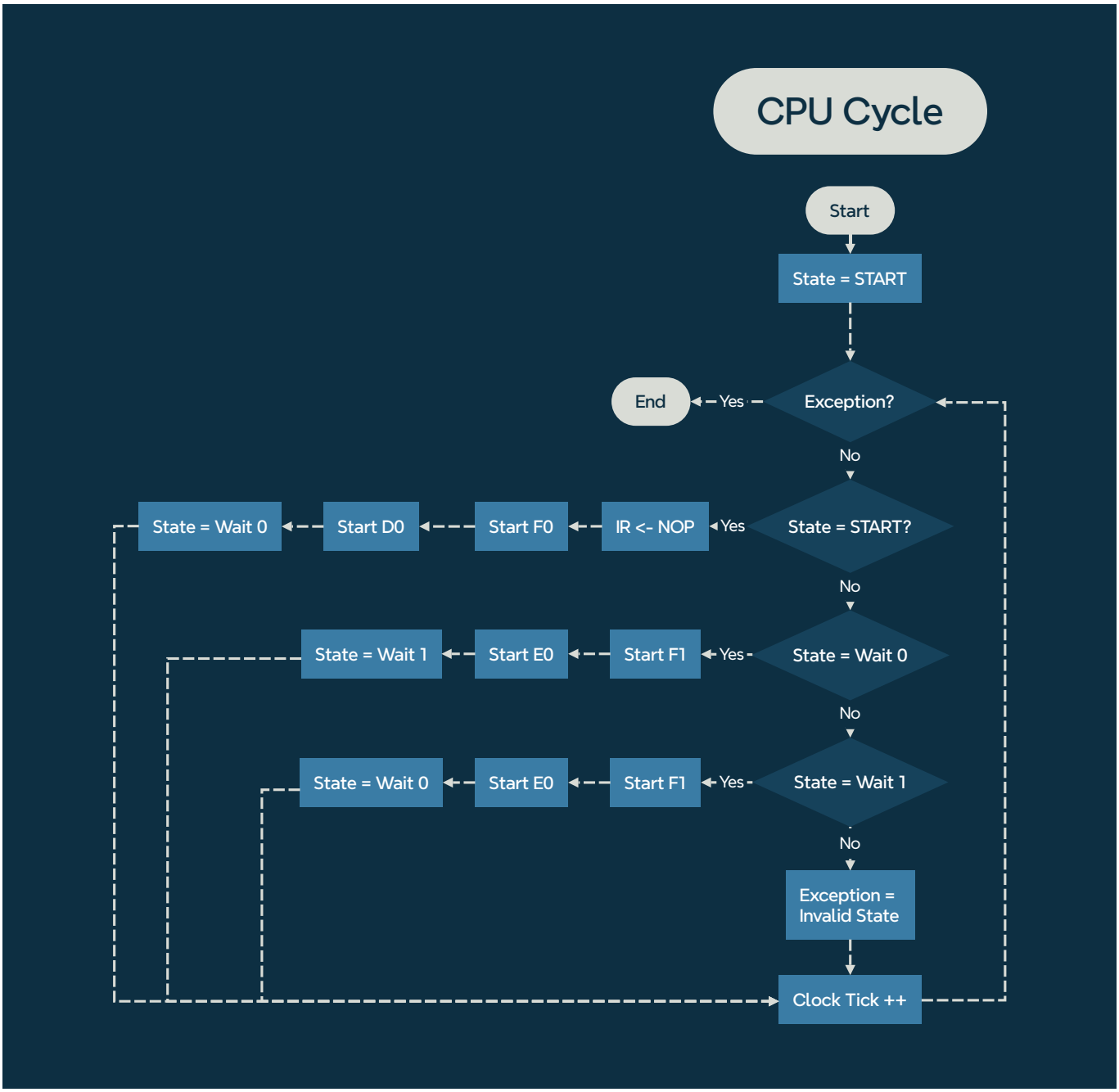
Zachary Fraser

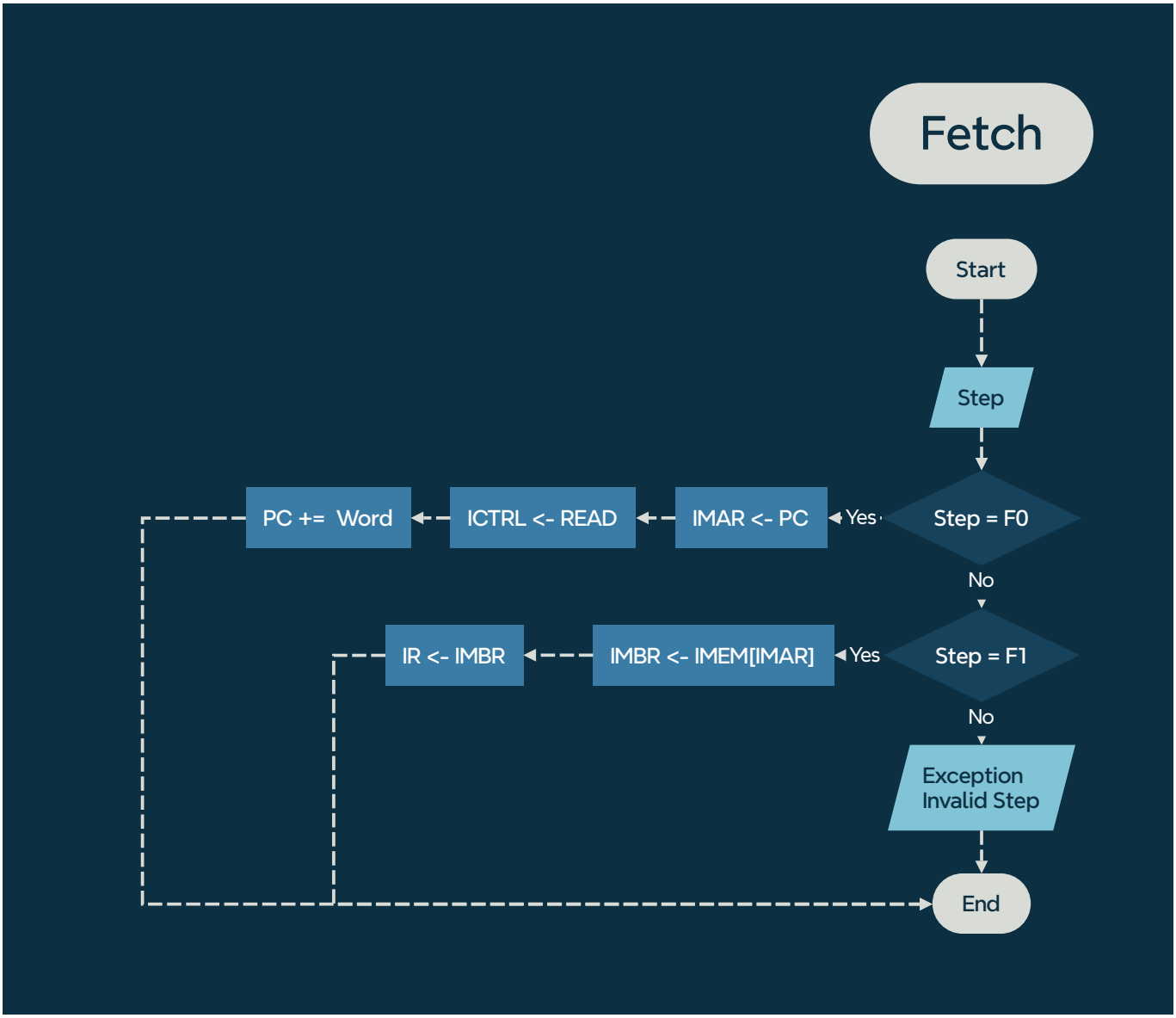
Assignment 2 - XM23P Instruction Pipeline

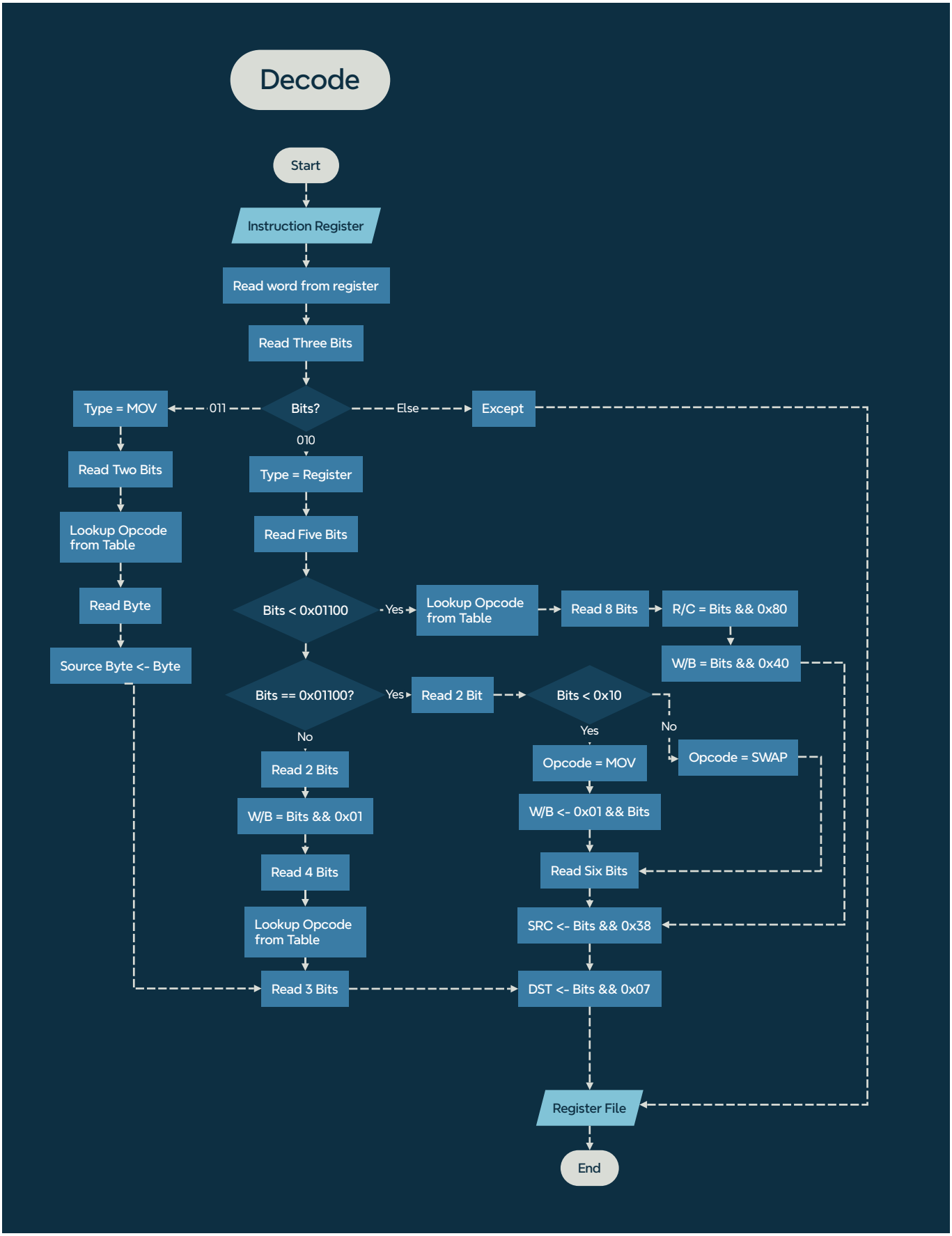
This assignment aims to implement the instruction pipeline for an XM23P emulator.

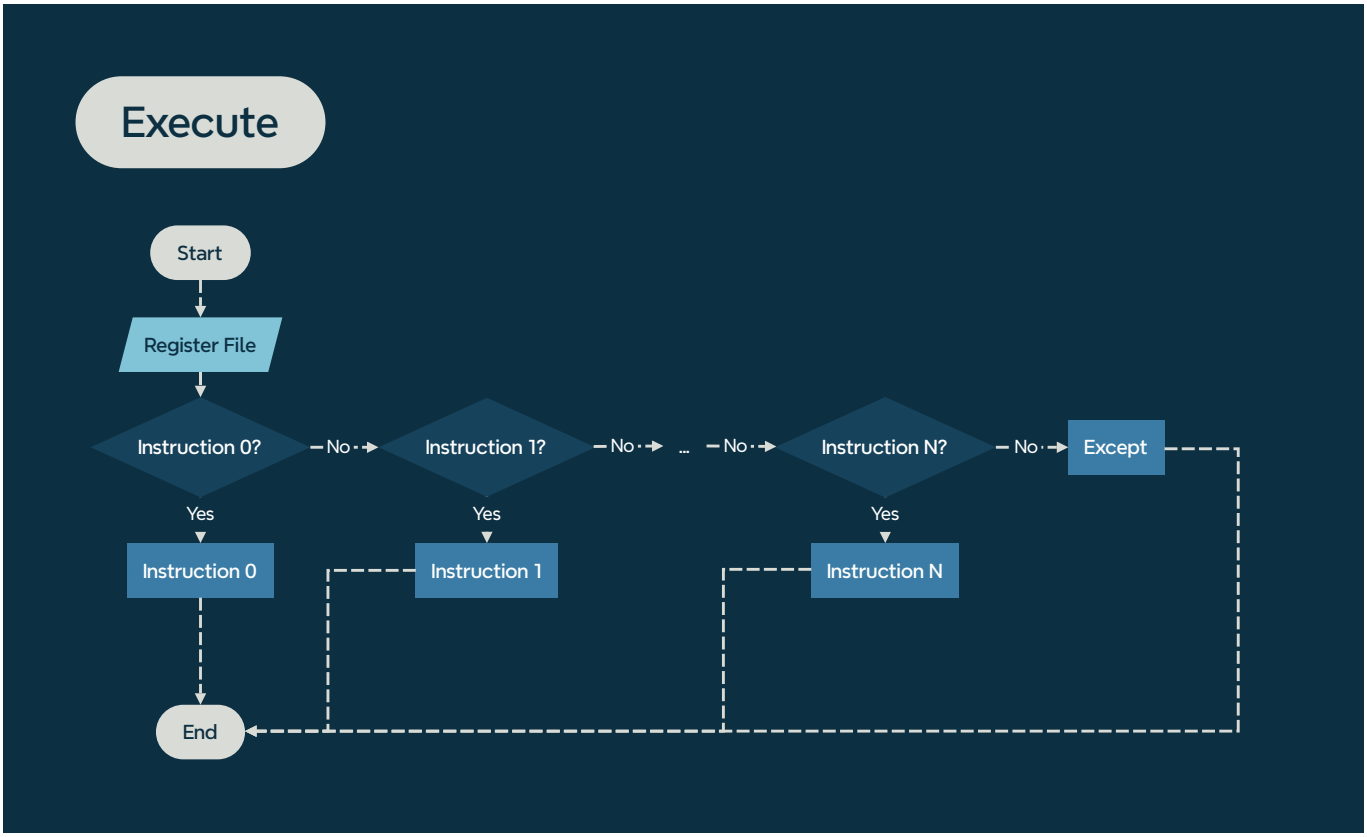
Design

The design contains logic flowcharts detailing the CPU Loop, and the Fetch, Decode, and Execute Stages. A Data dictionary describing the instructions and register file is also included.




Fetch
Start
Step
Step = F0
Yes
IMAR <- PC
ICTRL <- READ
PC += Word
No
Step = F1
Yes
IMBR <- IMEM[IMAR]
IR <- IMBR
No
Exception
Invalid Step
End


Decode
Start
Instruction Register
Read word from register
Read Three Bits
Bits?
011
Type = MOV
Else
Except
010
Type = Register
Read Two Bits
Lookup Opcode from Table
Read Byte
Source Byte <- Byte
Read Five Bits
Bits < 0x01100
Yes
Lookup Opcode from Table
Read 8 Bits
R/C = Bits && 0x80
W/B = Bits && 0x40
Bits == 0x01100?
Yes
Read 2 Bit
Bits < 0x10
Yes
No
Opcode = MOV
Opcode = SWAP
No
Read 2 Bits
W/B = Bits && 0x01
Read 4 Bits
Lookup Opcode from Table
W/B <- 0x01 && Bits
Read Six Bits
SRC <- Bits && 0x38
Read 3 Bits
DST <- Bits && 0x07
Register File
End


Execute
Start
Register File
Instruction 0?
No
Instruction 1?
No
...
No
Instruction N?
No
Except
Yes
Yes
Yes
Instruction 0
Instruction 1
Instruction N
End

Data Dictionary

```
INSTRUCTION     =   CODE + 1{PARAMETER}4
CODE            =   [0-20] *Contiguous encoding of instructions*
PARAMETER       =   [RC|WB|SOURCE|DESTINATION|BYTE]

RC              =   BIT
WB              =   BIT
SOURCE          =   3{BIT}3
DESTINATION     =   3{BIT}3

IMEM            =   32*2^10{WORD}32*2^10
IMAR            =   ADDRESS
ICTRL           =   [READ|WRITE]
IMBR            =   WORD
IR              =   WORD

ADDRESS         =   WORD
READ            =   0x0000
WRITE           =   0x0001
WORD            =   2{BYTE}2
BYTE            =   8{BIT}8
BIT             =   [0|1]
```