Kevin Martin
Syracuse University
CIS655 – Summer 2020, Tuesday @ 9:00pm EST
Homework 2

Question 2

For my IDE, I set it up using Microsoft Excel. As a quick background, I had worked in finance for about eight years before starting at Syracuse, and first learned basic programming concepts in visual basic (technically VBA, visual basic for applications). As such, I spent quite a bit of time in Excel throughout those years and thought it would be easy to make a nice looking and functional display. The result uses a combination of formulas and macros to achieve a user-friendly design.

The "language" supported here is a simplified version of MIPS. Each instruction is comprised of a 22 bit word: two bits for the type identification (I use 4 types as opposed to MIPS's 3 types), five bits for the instructions, and then five more bits for (up to) three registers. Each Opcode is a unique string and identifiable on its own, and each register has its own unique string of five bits as well. Therefore, no two instructions will be the same, unless it is asking for literally the exact same thing. The "Manual" tab has the description and related Opcode identifiers for each of the 47 instructions supported. There are 16 available registers, also on the Manual tab. Also print statements need to have each word separated by a semicolon as opposed to a space, however register values can be printed at will. Instructions that begin with a "#" are ignored completely. Array indices are stored as an integer value in each register, and you access the array followed by an offset to get the exact element of the array desired.

The "IDE" itself is on the IDE tab, and it also follows a similar outlay to MARS. The user only needs to use the whitespace provided in the "Edit" box, and the four buttons on the left-hand side of the screen. If the user needs more lines than provided by the whitespace, after the word "Continue" the gray space will automatically change to white to allow for more visually appealing typing. As far as the Data, this is simply comprised of 16 spaces in Memory, each with 7 four-bit values (identical to MARS here). Unfortunately, if the user needs more than those 16 spaces, memory is not available. Data is indexed using integer numbers during calculation, but is displayed using hexadecimal (again, following MARS).

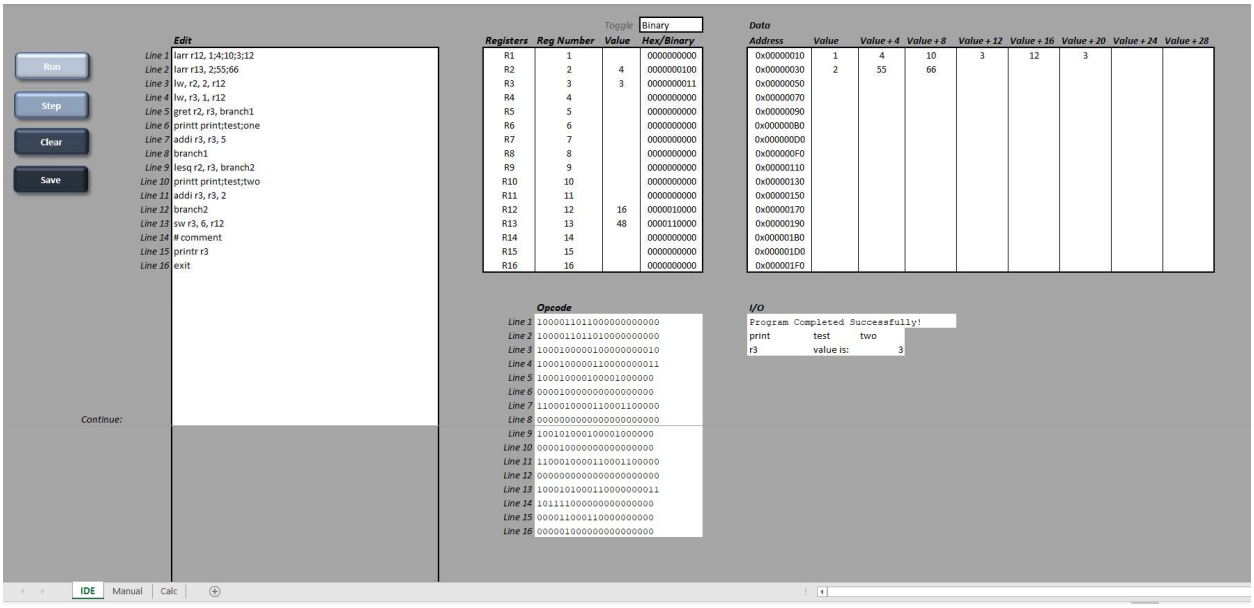
The four boxes on to the right (Registers, Data, Opcode and I/O) all update automatically. To use, simply enter the commands, and click "Run". The "Clear" button is helpful to see a fresh page, but it is not necessary to get an accurate line. "Save" simply saves the workbook according to regular Microsoft Office Save functionality (same as pressing Ctrl. + S on the keyboard). The Opcode section automatically translates each line in the edit window into its 22-bit word instruction. This section also changes from gray to white as needed. If the user chooses to enter any I/O (either print text or print register values), it will show up under the I/O section. Regardless of whether or not the user does use I/O, the phrase "Program Completed Successfully!" will be printed in that section as well.

Finally, note the "Step" button. I am pleased with how this one works as it was sort of a nice to have feature I initially thought of. Step does the same thing as Run, except it pauses at each instruction to allow the user to observe the register and data updates. After the user presses "Next", the next instruction is automatically updated.

For the screenshots, first the IDE tab where the user interacts with the program. Note that there are many formulas on this tab, the user does not need to be aware of any of them. Next, the Manual tab,

with the 47 instructions available and their related codes. Finally, there is a hidden “Calc” tab which is used during program execution. This tab remains hidden, but I am including in the screenshots just to show the full path. Some of the formulas on the first tab were easier to display with this extra tab, hence why it is needed. All of the buttons are supported by macros written in VBA, so I will include all of those as well. The macros interact with the sheet and each other to form the complete IDE.

IDE tab:



Manual page 1:

Description	Item	Type	Number	Code	OpCode	Operation														
																two bit type	2	4		
																4 bit opcode	5	32		
add	add	11	1	00001	1100001	+		R1	1	00001						4 bit register	5	32		
add immediately	addi	11	2	00010	1100010	+		R2	2	00010						3 registers	15			
subtract	sub	11	3	00011	1100011	-		R3	3	00011						**Total**	**22**	**bit words**		
subtract immediately	subi	11	4	00100	1100100	-		R4	4	00100										
multiply	mult	11	5	00101	1100101	*		R5	5	00101										
divide	div	11	6	00110	1100110	/		R6	6	00110										
modulo	mod	11	7	00111	1100111	MOD(		R7	7	00111										
power	pow	11	8	01000	1101000	POWER(		R8	8	01000										
square root	sqrt	11	9	01001	1101001	SQRT(		R9	9	01001										
square root of number times pi	sqpi	11	10	01010	1101010	SQRTPI(		R10	10	01010										
bitwise and	and	11	11	01011	1101011	BITAND(		R11	11	01011										
bitwise or	or	11	12	01100	1101100	BITOR(		R12	12	01100										
bitwise xor	xor	11	13	01101	1101101	BITXOR(		R13	13	01101										
bitwise shift left	bitl	11	14	01110	1101110	BITLSHIFT(		R14	14	01110										
bitwise shift right	bitr	11	15	01111	1101111	BITRSHIFT(		R15	15	01111										
multiply immediately	multi	11	16	10000	1110000	*		R16	16	10000										
divide immediately	divi	11	17	10001	1110001	/														
modulo immediately	modi	11	18	10010	1110010	MOD(														
power immediately	powi	11	19	10011	1110011	POWER(														
square root immediately	sqrti	11	20	10100	1110100	SQRT(														
square root of number times pi immediately	sqpii	11	21	10101	1110101	SQRTPI(														
bitwise and immediately	anii	11	22	10110	1110110	BITAND(														
bitwise or immediately	orii	11	23	10111	1110111	BITOR(														
bitwise xor immediately	xori	11	24	11000	1111000	BITXOR(														
bitwise shift left immediately	bilti	11	25	11001	1111001	BITLSHIFT(														
bitwise shift right immediately	bitri	11	26	11010	1111010	BITRSHIFT(														
		11	27	11011	1111011															
		11	28	11100	1111100															
		11	29	11101	1111101															
		11	30	11110	1111110															
		11	31	11111	1111111															
equal to	eqto	01	1	00001	100001	=														
graeter than	gret	01	2	00010	100010	>														
greater than or equl to	greq	01	3	00011	100011	>=														
less than	lest	01	4	00100	100100	<														
less than or equal to	lesq	01	5	00101	100101	<=														

Manual page 2:

38	*greater than or equl to*	greq	01	3	00011	100011	>=									
39	*less than*	lest	01	4	00100	100100	<									
40	*less than or equal to*	lesq	01	5	00101	100101	<=									
41	*not equal*	noeq	01	6	00110	100110	<>									
42	*equal to immediately*	eqtoi	01	7	00111	100111	=									
43	*greater than immediately*	greti	01	8	01000	101000	>									
44	*greater than or equal to immediately*	greqi	01	9	01001	101001	>=									
45	*less than immediately*	lesti	01	10	01010	101010	<									
46	*less than or equal to immediately*	lesqi	01	11	01011	101011	<=									
47	*not equal immediately*	noeqi	01	12	01100	101100	<>									
48	*jump to a specific input line*	jmpl	01	13	01101	101101										
49	*jump to a defined instruction*	jmpi	01	14	01110	101110										
50	*comment ignore*	#	01	15	01111	101111										
51			01	16	10000	110000										
52			10	1	00001	1000001										
53			10	2	00010	1000010										
54	*load array*	larr	10	3	00011	1000011										
55	*load item from word*	lw	10	4	00100	1000100										
56	*store item in word*	sw	10	5	00101	1000101										
57			10	6	00110	1000110										
58			10	7	00111	1000111										
59			10	8	01000	1001000										
60			10	9	01001	1001001										
61			10	10	01010	1001010										
62			10	11	01011	1001011										
63			10	12	01100	1001100										
64			10	13	01101	1001101										
65			10	14	01110	1001110										
66			10	15	01111	1001111										
67			10	16	10000	1010000										
68	*terminate gracefully*	exit	00	1	00001	000001										
69	*print text*	printt	00	2	00010	000010										
70	*print register direct*	printr	00	3	00011	000011										
71			00	4	00100	000100										
72			00	5	00101	000101										
73			00	6	00110	000110										
74			00	7	00111	000111										
75			00	8	01000	001000										
76			00	9	01001	001001										
77			00	10	01010	001010										

Calc tab (note: hidden at all times, only shown here for display purposes):

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	*Command*	*Dest*	*Reg 1*	*Reg 2*		*Operation*	*Dest Value*	*Reg 1 Value*	*Reg 2 Value*	*Opcode*	*Result*	*Truth*					
3	larr	r12	1;4;10;3;12			0	16	1;4;10;3;12	0	10							
4	larr	r13	2;55;66			0	48	2;55;66	0	10							
5	lw	r2	2	r12		0	4	2	16	10	2016	402					
6	lw	r3	1	r12		0	3	1	16	10	1016	301					
7	gret	r2	r3	branch1		>	4	3	branch1	1		TRUE					
8	printt	print;test;one				0	print;test;or	0	0	0	0						
9	addi	r3	r3	5		+	3	3	5	11	8	6					
10	branch1						0	0	0	999	0	0					
11	lesq	r2	r3	branch2		<=	4	3	branch2	1		FALSE					
12	printt	print;test;two				0	print;test;tv	0	0	0	0				print	test	two
13	addi	r3	r3	2		+	3	3	2	11	5	6					
14	branch2						0	0	0	999	0	0					
15	sw	r3	6	r12		0	3	6	16	10	6016	306					
16	#	comment				0	comment	0	0	1	0						
17	printr	r3				0	3	0	0	0	0	300					
18	exit					0	0	0	0	0	0	0					
19																	
20																	
21																	
22																	
23																	
24																	

VBA/Macro buttons:

```
Sub step()

Cells(16, 1).Value = 1

Call run_all

Cells(16, 1).Value = 0

End Sub
```

```
Sub run_all()

    If Cells(16, 1).Value = 1 Then
    Application.ScreenUpdating = True
    Else
    Application.ScreenUpdating = False
    End If

    Call clear_range
    Call text2col
    Call decode

    Cells(23, 13).Value = "Program Completed Successfully!"

    Application.ScreenUpdating = True

End Sub
```

```
Sub clear_range()

Application.ScreenUpdating = False

Dim clr_range As Range, clr2 As Range, clr3 As Range
Worksheets("Calc").Activate
Set clr_range = Worksheets("Calc").Range(Cells(3, 1), Cells(1000, 20))
clr_range.ClearContents
Worksheets("IDE").Activate
Set clr2 = Worksheets("IDE").Range(Cells(4, 14), Cells(19, 21))
clr2.ClearContents
Worksheets("IDE").Range(Cells(4, 9), Cells(19, 9)).Select
Selection.ClearContents
Cells(23, 13).Value = "** Press Run to Start **"
Set clr3 = Worksheets("IDE").Range(Cells(24, 13), Cells(200, 26))
clr3.ClearContents

Worksheets("IDE").Activate
ActiveSheet.Cells(1, 1).Select
Application.ScreenUpdating = True


End Sub
```

```
Sub text2col()

Application.DisplayAlerts = False
Application.EnableEvents = False
Dim dest As Range, inpt As Range, mv_range As Range, LastRow As Long

Set inpt = Worksheets("IDE").Range("E4", Range("E4").End(xlDown))
Set dest = Worksheets("Calc").Cells(3, 1)

inpt.TextToColumns DataType:=xlDelimited, consecutivedelimiter:=True, comma:=True, Space:=True, semicolon:=False, Destination:=Range("W1")
LastRow = Worksheets("IDE").Cells(Worksheets("IDE").Rows.Count, "W").End(xlUp).Row
Set mv_range = Worksheets("IDE").Range(Cells(1, 23), Cells(LastRow, 26))
mv_range.Select
Selection.Copy dest
mv_range.ClearContents

End Sub
```

```
Sub decode()

Worksheets("Calc").Activate
Dim code_range As Range, cell As Range, match_val As Variant, match_range As Range, var As Range, result_val As Long, var_row As Variant, _
arr_range As Range, arr_copy As Range, match_val2 As Variant, var2 As Range, offset_check As Integer, match_range2 As Range, _
match_val3 As Variant, var3 As Range, match_val4 As Variant, var4 As Range, data_int As Integer, data_int2 As Integer, usr_off As Integer, _
branch_val As Variant, branch_range As Range, i_val As Integer, print_range As Range, LastRow As Long, LastCol As Long, cpy_range As Range, print_row As Integer
offset_check = 16
print_row = 24
Worksheets("IDE").Activate
Set match_range = Worksheets("IDE").Range(Cells(4, 7), Cells(19, 7))
Set match_range2 = Worksheets("IDE").Range(Cells(4, 12), Cells(19, 12))
Worksheets("Calc").Activate
Set code_range = Worksheets("Calc").Range("A1", Range("A1").End(xlDown))
Range(Cells(2, 6), Cells(code_range.Rows.Count, 12)).FillDown
For i = 3 To code_range.Rows.Count:
Worksheets("Calc").Activate
    ' exit
    If Cells(i, 1).Value = "exit" Then
        MsgBox "Program Exited Successfully"
        Exit For

    '11 arithmatic
    ElseIf Cells(i, 10).Value = 11 Then
        result_val = Worksheets("Calc").Cells(i, 11).Value
        match_val = Worksheets("Calc").Cells(i, 2).Value
        Set var = match_range.Find(match_val, LookIn:=xlValues)
        var.Offset(0, 2).Value = result_val

    '01 branches
    ElseIf Cells(i, 10).Value = 1 And Cells(i, 12) = True Then
        branch_val = Cells(i, 4).Value
        Set branch_range = Range(Cells(3, 1), Cells(1000, 1))
        Set var = branch_range.Find(branch_val, LookIn:=xlValues)
        i = var.Row
```

```
    ElseIf Cells(i, 10).Value = 1 And Cells(i, 1) = "jmpi" Then
        branch_val = Cells(i, 2).Value
        Set branch_range = Range(Cells(3, 1), Cells(1000, 1))
        Set var = branch_range.Find(branch_val, LookIn:=xlValues)
        i = var.Row

    ElseIf Cells(i, 10).Value = 1 And Cells(i, 1) = "jmpl" Then
        jmp_line = Cells(i, 2).Value
        i = jmp_line + 3
        MsgBox i

    '10 arrays/memory
    ElseIf Cells(i, 1).Value = "larr" Then:
        Set arr_range = Worksheets("Calc").Cells(i, 3)
        arr_range.TextToColumns DataType:=xlDelimited, semicolon:=True, Destination:=Cells(i, 15)
        Worksheets("IDE").Activate
        Set var2 = match_range2.Find(offset_check, LookIn:=xlValues)
        Set paste_range = var2.Offset(0, 2)
        Worksheets("Calc").Activate
        Set arr_copy = Worksheets("Calc").Range(Cells(i, 15), Cells(i, 22))
        arr_copy.Copy
        Worksheets("IDE").Activate
        paste_range.PasteSpecial Paste:=xlPasteValues
        arr_copy.Clear
        match_val3 = Worksheets("Calc").Cells(i, 2).Value
        Set var3 = match_range.Find(match_val3, LookIn:=xlValues)
        var3.Offset(0, 2).Value = offset_check
        offset_check = offset_check + 32

    ElseIf Cells(i, 1).Value = "lw" Then
        usr_off = Worksheets("Calc").Cells(i, 3).Value
        match_val4 = Worksheets("Calc").Cells(i, 4).Value
        Set var4 = match_range.Find(match_val4, LookIn:=xlValues)
        data_int = var4.Offset(0, 2).Value
        Worksheets("IDE").Activate
        Set var3 = match_range2.Find(data_int, LookIn:=xlValues)
        data_int2 = var3.Offset(0, usr_off + 1).Value
        match_val3 = Worksheets("Calc").Cells(i, 2).Value
        Set var2 = match_range.Find(match_val3, LookIn:=xlValues)
        var2.Offset(0, 2).Value = data_int2

    ElseIf Cells(i, 1).Value = "sw" Then
        match_val3 = Worksheets("Calc").Cells(i, 2).Value
        Set var2 = match_range.Find(match_val3, LookIn:=xlValues)
        data_int2 = var2.Offset(0, 2).Value
        usr_off = Worksheets("Calc").Cells(i, 3).Value
        match_val4 = Worksheets("Calc").Cells(i, 4).Value
        Worksheets("IDE").Activate
        Set var4 = match_range.Find(match_val4, LookIn:=xlValues)
        data_int = var4.Offset(0, 2).Value
        Set var3 = match_range2.Find(data_int, LookIn:=xlValues)
        var3.Offset(0, usr_off + 1).Value = data_int2
```

```
    'll sys calls/misc
    ElseIf Cells(i, 1).Value = "printt" Then
        Worksheets("IDE").Activate
        Set print_range = Worksheets("IDE").Cells(print_row, 13)
        Worksheets("Calc").Activate
        Set arr_range = Worksheets("Calc").Cells(i, 2)
        arr_range.TextToColumns DataType:=xlDelimited, semicolon:=True, Destination:=Cells(i, 15)
        LastCol = Worksheets("Calc").Cells(i, Worksheets("Calc").Columns.Count).End(xlToLeft).Column
        Set cpy_range = Worksheets("Calc").Range(Cells(i, 15), Cells(i, LastCol))
        cpy_range.Copy 'print_range
        Worksheets("IDE").Activate
        print_range.PasteSpecial Paste:=xlPasteValues
        print_row = print_row + 1

    ElseIf Cells(i, 1).Value = "printr" Then
        match_val = Worksheets("Calc").Cells(i, 2).Value
        Worksheets("IDE").Activate
        Set var = match_range.Find(match_val, LookIn:=xlValues)
        result_val = var.Offset(0, 2).Value
        Cells(print_row, 13).Value = match_val
        Cells(print_row, 14).Value = "value is: "
        Cells(print_row, 15).Value = result_val
        print_row = print_row + 1

    'ElseIf Cells(i, 1).Value = "printw" Then
    '    match_val = Worksheets("Calc").Cells(i, 2).Value
    '    Worksheets("IDE").Activate
    '    Set var = match_range.Find(match_val, LookIn:=xlValues)
    '    result_val = var.Offset(0, 2).Value

    '   Cells(print_row, 13).Value = match_val
    '    Cells(print_row, 14).Value = "value is: "
    '    Cells(print_row, 15).Value = result_val

    Else:
        ' MsgBox "Please enter a valid operation"

    End If

If Cells(16, 1).Value = 1 Then
MsgBox "Press Next to Continue"
End If

Next i

Worksheets("IDE").Activate
ActiveSheet.Cells(1, 1).Select

End Sub
```

```
Sub save_wb()

ActiveWorkbook.Save

End Sub
```

One small user defined function for parsing input text, as part of the macros:

```
Function EvaluateString(strTextString As String)
Application.Volatile
EvaluateString = Evaluate(strTextString)
End Function
```