Software for the AIM 65

Unravel some mysteries of the AIM 65's monitor subroutines.

John D. Williams
8832 Boehning Ct.
Indianapolis IN 46219

With the wide variety of single-board computers on the market today, you might think that the introduction of yet another is hardly needed. But the AIM 65 by Rockwell fills a gap left by other single-board computers. Its many features provide capabilities beyond those of the usual single-board systems, yet it costs little more.

The AIM 65 is a versatile microcomputer. It is a 6502-based system with ASCII keyboard, 20 column thermal printer and 20 character alphanumeric display. The basic unit includes an 8K monitor/miniassembler, two parallel I/O ports and 1K of 2114 RAM. An additional 3K RAM, assembler and 8K Microsoft BASIC are available as on-board options.

Though little has been written about this system, it should be given serious consideration by anyone wishing to purchase a single-board computer. The purpose of this article is to briefly discuss the hardware aspects of the AIM 65 and then unravel a few of the mysteries surrounding the use of the monitor subroutines in programming. I hope it will also provide some insights into the use of the two parallel ports for control or interface applications.

Introduction

The AIM 65 has two buses: one is an expansion bus for adding additional memory; the other is an application bus. Part of this bus consists of two 8-bit parallel I/O ports with four control lines. These ports are controlled by a 6522 VIA.

The VIA (versatile interface adapter) also has an 8-bit shift register to convert data between parallel and serial formats, interrupt logic that can be polled and two 16-bit timers that can produce a single pulse or a series of pulses. One of the timers will be used as the basis of the real-time clock in the program. The parallel ports will provide the interface to the outside world.

The program was written to read voltage from a Valhalla True RMS Wattmeter. It will record the changes in voltage and the time of those changes. But its real usefulness lies in demonstrating how you can use several unique features of the AIM in ways not mentioned in the *User's Manual*.

```
<G>/
THE VOLTAGE IS
      VRMS-60HZ.
THE TIME IS
THIS PROGRAM WILL
MONITOR LINE VOLTAGE
AND RECORD THE
CHANGES.  IT WILL
ALSO RECORD THE TIME
OF THE CHANGES.
  TO INITIALIZE THE
PROGRAM, ENTER THE
TIME IN THIS FASHION
     HR:MIN:SEC.
ALLOW TWO DIGITS PER
UNIT.  THEN HIT F2 TO
START THE PROGRAM AT
THE RIGHT TIME.
```

Listing 1. Program text.

Specifically, the program uses the monitor's text editor to store the original prompt and subsequent statements used in the printout. It also uses both parallel I/O ports to specify what information is wanted and then to collect it. Finally, a real-time clock is used with one of the VIA's timers as its basis.

Monitor Subroutines

Listing 1 shows the text that is used. The text editor is used to store this section of the program. This feature is more convenient than stuffing consecutive memory locations with the proper ASCII codes. Simply begin the text buffer at address $0700. Be sure to hit return twice after all of the text is entered. This inserts $00 at the end of the text to signal the end of the text. It also stores the end text address in locations $00E1-00E2.

You may also want to employ one of the user-function keys to run the program. Program JMP 0400 at location $010C and JMP 0200 at location $010F. The program will begin running when you hit the F1 key. The F2 key starts the clock running and begins the voltage-monitoring section of the program. If the program is to be stored on tape, be sure to store addresses $010C-$0112 on the tape also. This allows the program to use F1 and F2 after being loaded from cassette tape.

The program in Listing 2 can be broken into four parts. Addresses $0200-0274 initialize the real-time clock and provide a printer and display for the clock. Locations $0300-033C contain the interrupt routine for the clock. The clock routine is similar to the one by Marvin L. DeJong in the March 1979 issue of *Micro*. The major change is in the clock display routine.

Locations $0400-04AD output the original prompt to the display and printer. They also allow you to input the time at which the main program will be started. Locations $0500-0593 monitor the voltage and output the change and time when the change occurred.

In the first section ($0200-0274), the first 18 instructions initialize the values for the 6522 VIA timer 1 interrupts. Locations $0215-021C store $C342 as the clock interrupt frequency. Locations $0226-0274 contain the display routine, which outputs the time to the display and printer. At location $0232 the X register is loaded with the value $0D. This is done because the value of the X register determines a character's position on the display. The actual display of the character is accomplished at location $024E by JSR E97A.

Location $E97A, an address in the AIM 65 monitor, is the start of a subroutine that takes an ASCII character in the accumulator and outputs it to the display and printer. The only problem is that the time is

Listing 2.

```
0200 78 SEI
0201 A9 LDA #00
0203 8D STA A404
0206 A9 LDA #03
0208 8D STA A405
020B A9 LDA #C0
020D 8D STA A00E
0210 A9 LDA #40
0212 8D STA A00B
0215 A9 LDA #42
0217 8D STA A006
021A A9 LDA #C3
021C 8D STA A005
021F A9 LDA #EC
0221 85 STA 00
0223 58 CLI
0224 00 BRK
0225 EA NOP
0226 A5 LDA 01
0228 85 STA 04
022A A5 LDA 02
022C 85 STA 05
022E A5 LDA 03
0230 85 STA 06
0232 A2 LDX #0D
0234 A0 LDY #06
0236 B9 LDA 0000,Y
0239 85 STA 10
023B 98 TYA
023C 85 STA 11
023E A0 LDY #04
0240 66 ROR 10
0242 88 DEY
0243 C0 CPY #00
0245 D0 BNE 0240
0247 A5 LDA 10
0249 29 AND #0F
024B 18 CLC
024C 69 ADC #30
024E 20 JSR E97A
0251 E8 INX
0252 A5 LDA 11
0254 A8 TAY
0255 B9 LDA 0000,Y
0258 29 AND #0F
025A 18 CLC
025B 69 ADC #30
025D 20 JSR E97A
0260 E8 INX
0261 88 DEY
0262 C0 CPY #03
0264 F0 BE0 026E
0266 A9 LDA #3A
0268 20 JSR E97A
026B 4C JMP 0236
026E 20 JSR EA13
0271 20 JSR EA13
0274 4C JMP 0500
0277 EA NOP
0300 48 PHA
0301 E6 INC 00
0303 D0 BNE 0337
0305 F8 SED
0306 18 CLC
0307 A5 LDA 01
0309 69 ADC #01
030B 85 STA 01
030D C9 CMP #60
030F 90 BCC 0313
0311 A9 LDA #00
0313 85 STA 01
0315 18 CLC
0316 A5 LDA 02
0318 69 ADC #01
031A 85 STA 02
031C C9 CMP #60
031E 90 BCC 0322
0320 A9 LDA #00
0322 85 STA 02
0324 18 CLC
0325 A5 LDA 03
0327 69 ADC #01
0329 85 STA 03
032B C9 CMP #24
032D 90 BCC 0331
032F A9 LDA #00
0331 85 STA 03
0333 A9 LDA #EC
0335 85 STA 00
0337 AD LDA A004
033A D8 CLD
033B 68 PLA
033C 40 RTI
033D EA NOP
0400 A2 LDX #00
0402 A0 LDY #00
0404 A9 LDA #00
0406 8D STA 04F0
0409 B9 LDA 072E,Y
040C C9 CMP #0D
040E F0 BE0 0418
0410 20 JSR E97A
0413 E8 INX
0414 C8 INY
0415 4C JMP 0409
0418 20 JSR EA13
041B A2 LDX #00
041D C8 INY
041E EE INC 04F0
0421 AD LDA 04F0
0424 C9 CMP #07
0426 D0 BNE 0409
0428 A0 LDY #00
042A B9 LDA 07BE,Y
042D C9 CMP #0D
042F F0 BE0 0439
0431 20 JSR E97A
0434 E8 INX
0435 C8 INY
0436 4C JMP 042A
0439 20 JSR EA13
043C C8 INY
043D A2 LDX #00
043F EE INC 04F0
0442 AD LDA 04F0
0445 C9 CMP #0E
0447 D0 BNE 042A
0449 20 JSR E95F
044C 8D STA 04F1
044F 20 JSR E95F
0452 8D STA 04F2
0455 20 JSR E95F
0458 8D STA 04F3
045B 20 JSR E95F
045E 8D STA 04F4
0461 20 JSR E95F
0464 8D STA 04F5
0467 20 JSR E95F
046A 8D STA 04F6
046D A2 LDX #01
046F A0 LDY #04
0471 3E ROL 04F0,X
0474 88 DEY
0475 C0 CPY #00
0477 D0 BNE 0471
0479 BD LDA 04F0,X
047C 29 AND #F0
047E 18 CLC
047F 9D STA 04F0,X
0482 E8 INX
0483 E8 INX
0484 E0 CPX #07
0486 D0 BNE 046F
0488 A2 LDX #02
048A BD LDA 04F0,X
048D 29 AND #0F
048F 18 CLC
0490 CA DEX
0491 7D ADC 04F0,X
0494 9D STA 04F0,X
0497 E8 INX
0498 E8 INX
0499 E8 INX
049A E0 CPX #08
049C D0 BNE 048A
049E AD LDA 04F1
04A1 85 STA 03
04A3 AD LDA 04F3
04A6 85 STA 02
04A8 AD LDA 04F5
04AB 85 STA 01
04AD 60 RTS
0500 A9 LDA #FF
0502 8D STA A002
0505 A9 LDA #00
0507 8D STA A003
050A A2 LDX #00
050C A0 LDY #00
050E A9 LDA #02
0510 8D STA 0625
0513 8D STA A000
0516 AD LDA A00F
0519 9D STA 0600,X
051C DD CMP 0650,X
051F D0 BNE 0542
0521 BD LDA 0600,X
0524 9D STA 0650,X
0527 E8 INX
0528 CE DEC 0625
052B AD LDA 0625
052E C9 CMP #00
0530 D0 BNE 0513
0532 8D STA A000
0535 AD LDA A00F
0538 9D STA 0600,X
053B DD CMP 0650,X
053E D0 BNE 0546
0540 F0 BE0 054D
0542 C8 INY
0543 4C JMP 0521
0546 C8 INY
0547 BD LDA 0600,X
054A 9D STA 0650,X
054D 98 TYA
054E C0 CPY #00
0550 F0 BE0 050A
0552 A2 LDX #00
0554 BD LDA 0700,X
0557 C9 CMP #0D
0559 F0 BE0 0562
055B 20 JSR E97A
055E E8 INX
055F 4C JMP 0554
0562 20 JSR EA13
0565 A2 LDX #00
0567 BD LDA 0650,X
056A 20 JSR E97A
056D E8 INX
056E E0 CPX #03
0570 D0 BNE 0567
0572 BD LDA 0713,X
0575 C9 CMP #0D
0577 F0 BE0 0580
0579 20 JSR E97A
057C E8 INX
057D 4C JMP 0572
0580 20 JSR EA13
0583 A2 LDX #00
0585 BD LDA 0721,X
0588 C9 CMP #0D
058A F0 BE0 0593
058C 20 JSR E97A
058F E8 INX
0590 4C JMP 0585
0593 4C JMP 0226
0596 EA NOP
```

stored in decimal form in the memory.

Before the characters can be displayed, they must be converted to the ASCII format. This is accomplished by first storing the two-digit number in an address where it can be manipulated. The number is rotated right four bits so that the most significant nibble (MSN) is now the least significant nibble (LSN). It is then ANDed with $0F to save only the LSN. $30 is then added to produce the correct ASCII code for any decimal number 0-9.

To display the second number of the pair, the original two digits are once again retrieved from memory. It is not necessary to rotate this data since the LSN is in the proper place. It is simply converted to ASCII in the same manner as the other and displayed. Between the hours, minutes and seconds, a colon is displayed.

After the time has been displayed, there is another jump to a monitor subroutine. The subroutine at $EA13 outputs a carriage return and line feed to the display and printer. This is done twice. The final instruction executes a jump to another section of the program, which looks for a change in voltage.

The second section, from locations $0300-033C, is the clock. This section is used every time an interrupt is generated by the timer in the VIA. When 24 hours are up, the clock resets itself and continues to run. The clock continues to keep fairly accurate time even while other parts of the program are being run.

The third section, locations $0400-04AD, outputs the original prompt and allows the user to input the time he wishes to start the program. The use of the text buffer as the source of the prompt is accomplished by using the X register as the display counter, the Y register as the text buffer location counter and $04F0 as a text line counter. The characters are loaded into the accumulator by using absolute indexed addressing.

The absolute address is the starting address of the text. The Y register is then added to the absolute address to obtain the correct address for the character to be displayed. The Y register is then incremented to obtain the consecutive characters. A carriage return (ASCII $0D) will be found at the end of each line.

When a CR is loaded into the accumulator, the program jumps to a subroutine that outputs a carriage return and line feed to the display and printer. The Y register is incremented to be ready to fetch the next character. The X register is set to zero in order that the next line of characters appears at the proper place on the display and printer. Address $04F0 is then incremented and tested to see if all the lines of the text have been output. If not, the program branches back to $042A, ready to fetch the next line. Otherwise, the program looks at the keyboard using another subroutine.

At this point, the user inputs the time he will start the program. The clock will use this time as its starting time. The hour, minute and second times are input as two-digit numbers (eg., 01:15:30). Since each number is in ASCII format as it is taken from the keyboard, the two numbers for each division must be converted to decimal form and combined into one byte. This is done at locations $046D-$049C.

The decimal numbers are then loaded into the proper addresses for use by the clock. The program then returns to the monitor. When the time input to the clock is reached, the user hits F2 and the program begins to run.

The final section ($0500-0599) reads data from the meter and outputs the changes to the display and printer. This is accomplished by using ports A and B of the 6522 VIA. As it is presently structured, the program will read three digits by outputting on port B which digit it wishes to look at. It then reads the data on port A. Both the digits read and those output are in BCD format.

The first four instructions initialize port B as an output port and port A as an input port. While this program initializes the entire port as one or the other, each individual bit can actually be selected as an input or output. The requested digit is output on port B by loading address $A000 with the requested digit.

The most significant digit is chosen first. It is then compared with the previous reading. If it is not the same, the reading is stored and the Y register is incremented.

After the three digits have been read, the Y register is checked to see if any digits have changed. If not, the process is started again. If the data has changed, more text is taken from the text buffer. It explains what the data is. The new voltage is then output to the display and printer. The clock is then checked, and the current time is also output. The process then begins again.

The simplicity of using the I/O ports should now be evident. All you have to do is to select the function of the port by loading the proper address with 1s or 0s. Even the individual bits can be independently programmed for function. The port can then be read by loading the accumulator with the value found at the port's address. Data can be output on a port by storing the data at the port's address. What could be simpler than that?

This program looks at a different digit approximately 8000 times per second. This means a complete reading is done 44 times per line cycle. The only drawback is that most digital voltmeters do not update their ouput nearly that fast. Your ability to use this program to record voltage fluctuations or line spikes will be limited by the meter used for the measurements.

Listing 3 is a sample run of the program. Note that when inputting the time, you do not separate the hours, minutes and seconds by colons. These are inserted by the program for the output. Although this program was run on an AIM 65 with 4K RAM, it is easily run in the 1K version. The only change involved is the reassigning of addresses.

```
<CR>THIS PROGRAM WILL
MONITOR LINE VOLTAGE
AND RECORD THE
CHANGES.  IT WILL
ALSO RECORD THE TIME
OF THE CHANGES.
  TO INITIALIZE THE
PROGRAM, ENTER THE
TIME IN THIS FASHION
     HR:MIN:SEC.
ALLOW TWO DIGITS PER
UNIT.  THEN HIT F2 TO
START THE PROGRAM AT
THE RIGHT TIME.
112220
<CR>11:22:20

THE VOLTAGE IS
042VRMS-60HZ.
THE TIME IS 11:22:22

THE VOLTAGE IS
046VRMS-60HZ.
THE TIME IS 11:22:31

THE VOLTAGE IS
074VRMS-60HZ.
THE TIME IS 11:22:33

THE VOLTAGE IS
120VRMS-60HZ.
THE TIME IS 11:22:36

THE VOLTAGE IS
114VRMS-60HZ.
THE TIME IS 11:22:38

THE VOLTAGE IS
117VRMS-60HZ.
THE TIME IS 11:22:40

THE VOLTAGE IS
115VRMS-60HZ.
THE TIME IS 11:22:42

THE VOLTAGE IS
117VRMS-60HZ.
THE TIME IS 11:22:44
```

Listing 3. Sample run.

```
0000 A9 LDA #00
0002 85 STA A0
0004 85 STA A2
0006 A9 LDA #02
0008 85 STA A1
000A A0 LDY #00
000C B1 LDA (A0),Y
000E C9 CMP #0D
0010 F0 BEQ 002F
0012 20 JSR E97A
0015 C0 CPY #FF
0017 F0 BEQ 001D
0019 C8 INY
001A 4C JMP 000C
001D 18 CLC
001E A5 LDA A0
0020 69 ADC #FF
0022 85 STA A0
0024 A5 LDA A1
0026 69 ADC #00
0028 85 STA A1
002A C8 INY
002B C8 INY
002C 4C JMP 000C
002F 20 JSR EA13
0032 E6 INC A2
0034 A5 LDA A2
0036 C9 CMP #10
0038 F0 BEQ 0042
003A C0 CPY #FF
003C F0 BEQ 001D
003E C8 INY
003F 4C JMP 000C
0042 EA NOP
0043 60 RTS
0044 EA NOP
```

Listing 4. Alternative method of text display.

Alternative Method

This program uses absolute indexed addressing to output text from the text buffer. The advantage of this method is that it takes few steps to accomplish. The drawback is that it can only make use of text no longer than 256 characters.

Listing 4 shows an alternative method of outputting text. It uses indirect indexed addressing. While it takes more steps to accomplish the same goal, it can address text of up to 64K characters in length.

This particular example starts the text address at $0200. This information is stored at locations $A0 and $A1. Address $A2 is used as the text line counter.

Once again it is the carriage return (ASCII $0D) that is used to detect the end of a line. The Y register is incremented to $FF and then added to the address found at $00A0 and $00A1. $00A0 contains the low-order byte of the text address, while $00A1 contains the high-order byte. Since the address at $A0 and $A1 can be changed, any length text at any location can be addressed. This method has been used successfully to print text over 1K in length.

This program uses many of the unique features of the AIM 65. The text editor, parallel I/O ports and the wide variety of monitor subroutines provide a combination of features not to be found anywhere else in the world of single-board computers. The AIM is easy to use once the user has unlocked a few of its secrets and is able to make use of the monitor subroutines and other features. I hope that this program has given you some new insights into programming the AIM 65. ■