Robert Rienstra WA8GRG
5877 Parliament Dr.
Columbus OH 43213

SWTP/H14 Get-Together

—a driver routine for the Heathkit line printer

Not too long ago, I purchased a Heathkit printer, Model H14. Anyone familiar with Heathkit's excellent documentation and instructions knows how easily he can become spoiled. This certainly affected my programming. I became careless. You might say that this is natural, but we programmers all know that this means sloppy programs. This came to light when I told my trustworthy SWTP 6800 computer to start sending data to my new printer.

```
                *** PRINTER DRIVER***
                *THIS SUBROUTINE WAS DEVELOPED TO SUPPORT
                *THE "HEATHKIT" LINE PRINTER.
                *AUTHOR:
                *ROBERT RIENSTRA
                *5877 PARLIAMENT DR.
                *COLUMBUS - OHIO 43213
                *
                *
                *STORAGE
A010            TEMPXR EQU   $A010     TEMPSTORAGE FOR 'X' REGISTER
A062            BYTCNT EQU   $A062     BYTE COUNT (16)
801C            PORT07 EQU   $801C     PORT ADDR. FOR PRINTER
A060            PRTTST EQU   $A060     TEST PORT TO SEE IF INITIALIZED
                *
                *
                *PRINT DRIVER SUBROUTINE
                *
C4A8                   ORG   $C4A8     PROM ADDRESS.
                       OPT   LIS,SYM
C4A8 37                PSH B           SAVE B ACCUM.
C4A9 FF A0 10          STX   TEMPXR    SAVE 'X' REGISTER
C4AC 8D 35             BSR   JMP1      JUMP TO DEVICE CLEARED AND INIT. LINK
C4AE E6 00      LOOP   LDA B 0,X       SEE IF DEVICE IS READY FOR DATA
C4B0 57                ASR B
C4B1 57                ASR B
C4B2 24 FA             BCC   LOOP      TEST AGAIN
C4B4 A7 01             STA A 1,X       OUTPUT ONE CHARACTER TO PRINTER
C4B6 8D 05             BSR   INTRPT    SEE IF PRINTER SENT INTERRUPT SIGNAL
C4B8 33                PUL B           RESTORE B ACCUM.
C4B9 FE A0 10          LDX   TEMPXR    RESTORE 'X' REGISTER
C4BC 39         EXIT00 RTS             EXIT DRIVER ROUTINE
                *
                *
                *THIS SUBROUTINE EXAMINES THE PRINTER'S
                *INTERRUPT SIGNALS.
                *
C4BD 7D A0 62   INTRPT TST   BYTCNT    IS BYTE COUNTER = ZERO
C4C0 26 17             BNE   ADD1
C4C2 A6 00      LOOP1  LDA A 0,X       TEST  IF HANDSHAKE SIGNAL IS PRESENT
C4C4 47                ASR A
C4C5 24 10             BCC   TIME1
C4C7 A6 01      CKINTR LDA A 1,X
C4C9 81 13             CMP A #$13       PRINTER BUFFER IS FULL
C4CB 27 05             BEQ   BYTE16
C4CD 81 11             CMP A #$11       PRINTER NEEDS MORE DATA
C4CF 26 F1             BNE   LOOP1
C4D1 39         EXIT01 RTS             EXIT INTERRUPT TEST
C4D2 86 0F      BYTE16 LDA A #$0F      LOAD "A" WITH 16 COUNT
C4D4 B7 A0 62          STA A BYTCNT     STORE IN COUNTER
C4D7 20 EE             BRA   CKINTR
C4D9 7A A0 62   ADD1   DEC   BYTCNT
C4DC 27 DF             BEQ   INTRPT
C4DE 39         EXIT02 RTS             EXIT
                *
                *
                *THIS IS THE LINK TO THE TIMING ADJUST ROUTINE
                *
C4DF 8D 00      TIME1  BSR   NEXT1     JUST TO STRETCH THE TIME
C4E1 20 74      NEXT1  BRA   BITDLY    GO TO BIT DELAY
C4E3 20 5B      JMP1   BRA   CLRDEV    CLEAR DEVICE AND INITIALIZE
                *
                *
                *THIS SUBROUTINE CLEARS AND INITIALIZES THE ACIA
                *
                *
C540                   ORG   $C540
```

Program listing.

I made sure that all the required paraphernalia were present: RS232C line, handshake line, serial I/O interface. I even came up with a program of sorts.

The Heathkit printer contains:

1. One 256 byte line buffer.

2. Handshake control signals. Hex 13 = Buffer full. Hex 11 = Send more data (16 bytes only). More about these control signals later.

3. Three different characters/inch print modes.

4. Print action started by either a "CR" or when any of the selected number of characters/inch modes are satisfied.

The handshake signals were giving me the most problems, as I did not know what actually took place. Hex 13 = buffer full means what it says and the program must stop sending data until hex 11 is received after which only 16 characters can be sent until another hex 11 is received. I found out by hooking up my scope that hex 11 is transmitted by the printer right after the matrix printhead has reached the end of the printline and starts its return trip. So, programmers be aware.

After many experiments and reprogramming sessions, I came up with a program which seemed to answer my problem. See the program listing.

The program was compiled with an EPROM starting address. The PORT address is the SWTP's printer port by convention. The X register and the B accumulator have been saved at the beginning and are restored at the end. The serial device ACIA is always initialized and cleared. This subroutine starts at location 'C540'. The 92-position opening between LOC. 'C4E5' and LOC. 'C540' is because of another printer-oriented subroutine residing there.

This program is by no means the ultimate answer, and I am sure that somewhere in this country somebody else may also have tackled the same program problems, but all I can say is that Heathkit's line printer runs smoothly and flawlessly at 4800

baud without missing one character. If you have to take the printer off line (out of paper, etc.), the computer will loop in the program and never lose one character.

Relocating the program to another location should be easy, as the program is completely relocatable. This program is a subroutine, so all you have to do is load the A accumulator with the ASCII character to be printed and execute this program using the BSR instruction (BSR #$C4A8).

Program Description

The program's main line starts at LOC. 'C4A8'; it ends at LOC. 'C4BC'. After the ACIA device has been tested to see if it is free to receive more data, accumulator A is stored in the device's data register '801D'. The program then performs a subroutine that tests the interrupt (handshake) codes (LOC. C4BD through C4DE).

This interrupt subroutine not only takes care of the interrupt codes, but also deals with an inherent timing problem automatically created by the moving printhead and data transmission. TAG "TIME1" and "NEXT1" will adjust this delay I mentioned. Did I say delay? I mean timing. TAG "BITDLY" adjusts the timing problem. Decreasing or increasing the 04 count may cause the interrupt subroutine to miss the handshake signals (hex 11 or hex 13) and, therefore, cause garbage to be printed.

TAG "CLRDEV" really does not need any explanation, as it initially clears and programs the ACIA device.

Now I'm looking for a way to use the SWTP CT-64 at a higher baud rate, say 4800.■

✓Reader Service—see page 226