can be seven or eight bits, and there may be a parity bit appended to that data. A parity bit is an extra bit that helps detect transmission errors. Finally, there may be one or two stop bits. The various options available are as follows:

Control Register			Number Of Data Bits	Parity	Number Of Stop Bits
Bit 4	Bit 3	Bit 2			
0	0	0	7	Even	2
0	0	1	7	Odd	2
0	1	0	7	Even	1
0	1	1	7	Odd	1
1	0	0	8	None	2
1	0	1	8	None	1
1	1	0	8	Even	1
1	1	1	8	Odd	1

The two least significant bits (0 and 1) are used to determine the speed of transmission and reception. This is done by setting a divisor for the clock rate. The 6850 does not have its own clock, and therefore must be provided with an external one, typically set at 1,760 Hz.

Control Register		Clock Rate
Bit 1	Bit 0	Divisor
0	0	1
0	1	16
1	0	64

The combination not shown in the table, when both these bits are one, causes a master reset of the chip.

In the status register the bits have the following functions:

Bit	Function
7	Interrupt request
6	Set if a parity error occurs on reception
5	Set if the receiver overruns, i.e. too many bits were received, with characters running into the previous ones
4	Set if a framing error occurs on reception — the wrong number of start and stop bits
3	Set when a signal is received on the CTS line
2	Set when a signal is received on the DCD line
1	Set when the transmit data register is empty
0	Set when the received data register is full

Our second example program uses a 6850 chip to receive a character string, terminated by a carriage return, from a remote terminal. The principle is to program the chip appropriately, then loop round checking if the receive data register is full. When it is, we remove the data byte, which resets bit 0 in the status register. The process is repeated until the character received is a carriage return (ASCII code 13). We shall be ignoring any transmission errors, though checking for them by masking the contents of the status register to see if any of the error indicating bits are set is quite straightforward. We shall assume a fairly common protocol: eight data bits, no parity and two stop bits and a divide by 16 clock speed. The first subroutine programs the chip, the second receives the data.

PIA Program

```
TABLE   EQU   $3000           Subroutine to set up Port A
        ORG   $1000

        ASLA                  Shift A left to multiply by two
                              (the table consists of two-byte
        LDX   #TABLE          addresses)
        LDX   A,X             Get base address of PIA
        CLR   1,X             Get access to data direction
        LDB   #%11111111      register
        STB   ,X              Set all bits for output
        LDB   #%00101100      Disable interrupts, set control
        STB   1,X             line 2 for output and select
        RTS                   data register
                              Subroutine to print string
                              whose address is in Y
        ASLA                  Shift A left to multiply by two
        LDX   #TABLE          Get base address of PIA
        LDX   A,X
        LDA   ,Y+             Get length of string in A
LOOP1   BEQ   FINISH          Check if length is zero
LOOP2   LDB   1,X             Check if ready for next bit
        ANDB  #%10000000      Mask off all except bit 7
        BEQ   LOOP2           If not ready
        LDB   ,Y+             Get next character
        STB   ,X              Print it
LOOP3   LDB   1,X             Check if transmitted
        ANDB  #%01000000      Look at bit 6
        BEQ   LOOP3           Loop if not ready yet
        LDB   ,X              Read data register to clear
                              status bits
        DECA                  Subtract one from length
        BRA   LOOP1           Get next character
FINISH  RTS
```

ACIA Program

```
TABLE   EQU   $3000           Subroutine to program 6850
        ORG   $1000
                              Subroutine to set up ACIA
ACIAST  ASLA                  Shift A left to multiply by two
                              (table of two-byte addresses)
        LDX   #TABLE          Get base address of ACIA
        LDX   A,X
        LDA   #%00000011      Master reset of ACIA
        STA   ,X              Into control register
        LDA   #%00010001      Program ACIA (8 data bits, no
                              parity, 2 stop bits)
        STA   ,X
        RTS                   Subroutine to accept string of
                              characters
BUFFER  EQU   $4000           Somewhere to put the string
CR      EQU   13              ASCII for carriage return
        BSR   ACIAST          Set up ACIA. X register
                              contains ACIA address
        LDY   #BUFFER         Destination in Y
LOOP    LDB   ,X              Get status
        ASLB                  Shifts bit 7 out of B register
                              into carry flag of CCR
        BCC   LOOP            Go back if that bit was not set,
                              i.e. there is no interrupt
                              request yet
        LDA   1,X             Get data byte
        STA   ,Y+             Store it
        CMPA  #CR             Is A a carriage return?
        BNE   LOOP            Next character
        RTS
```