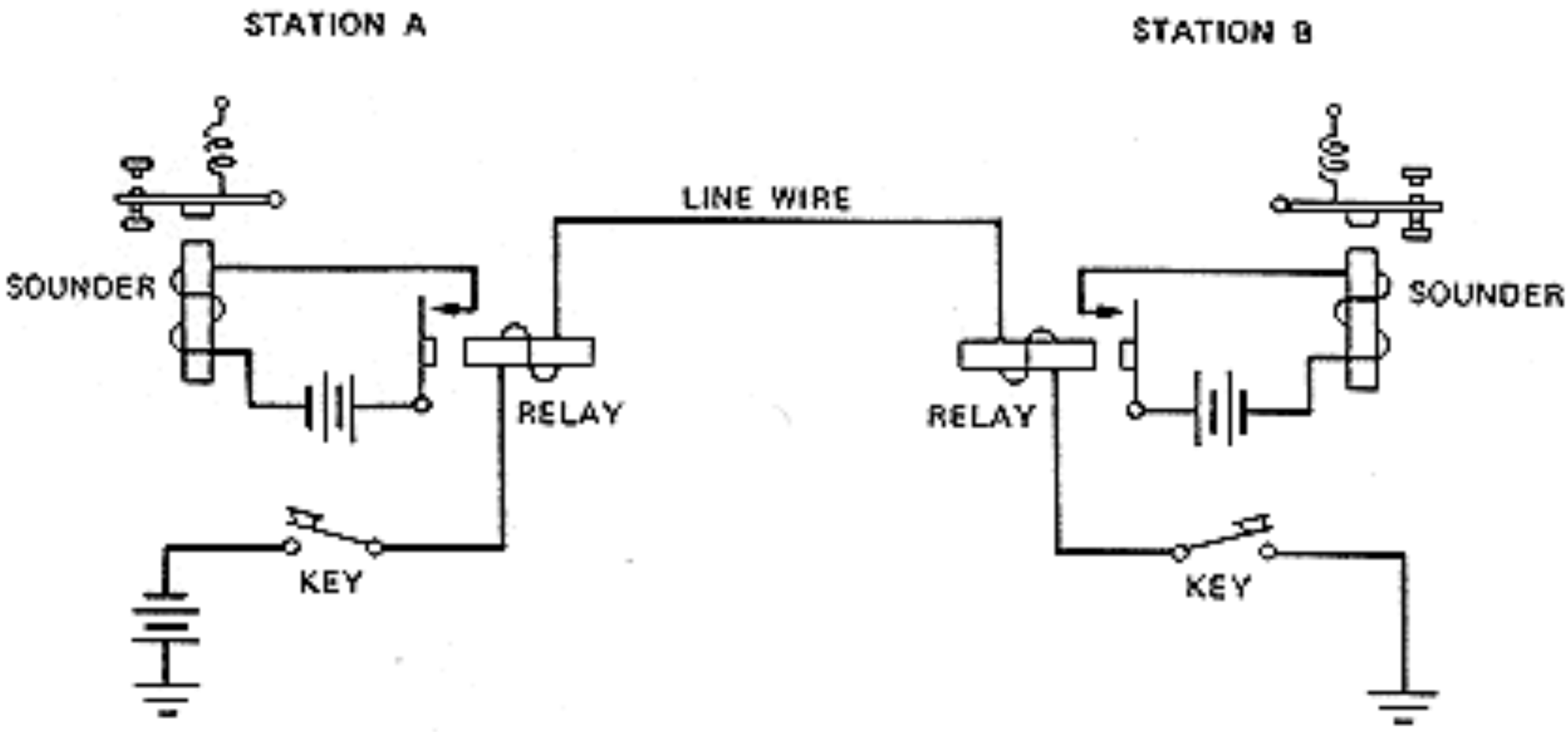
Communication

The Serial Protocol
and
ASCII Character Codes

SIMPLEX TELEGRAPH

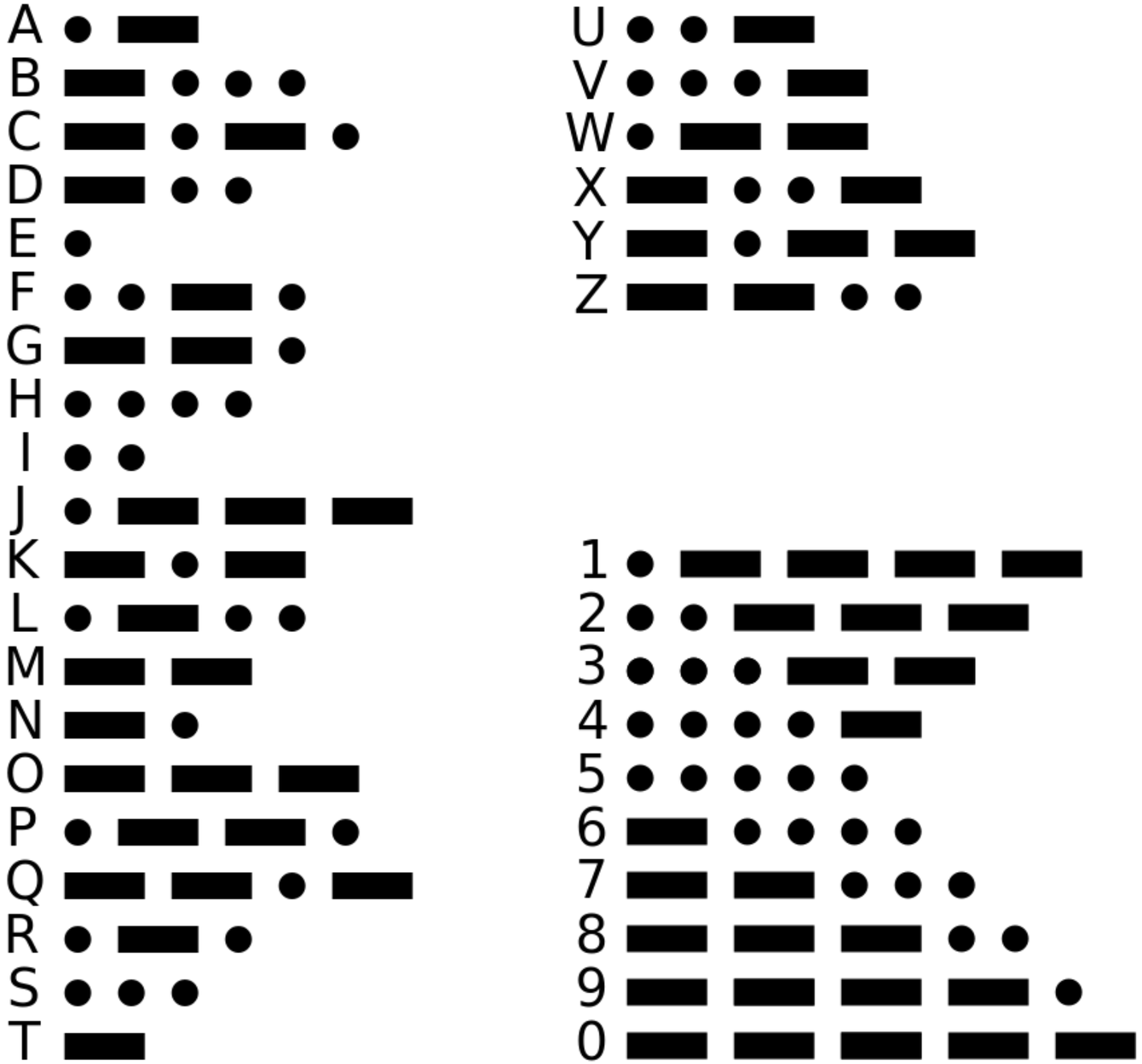


Elementary neutral telegraph circuit.

http://people.seas.harvard.edu/~jones/cscie129/nu_lectures/lecture5/elecmag_tel/morse_tel.html

International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.



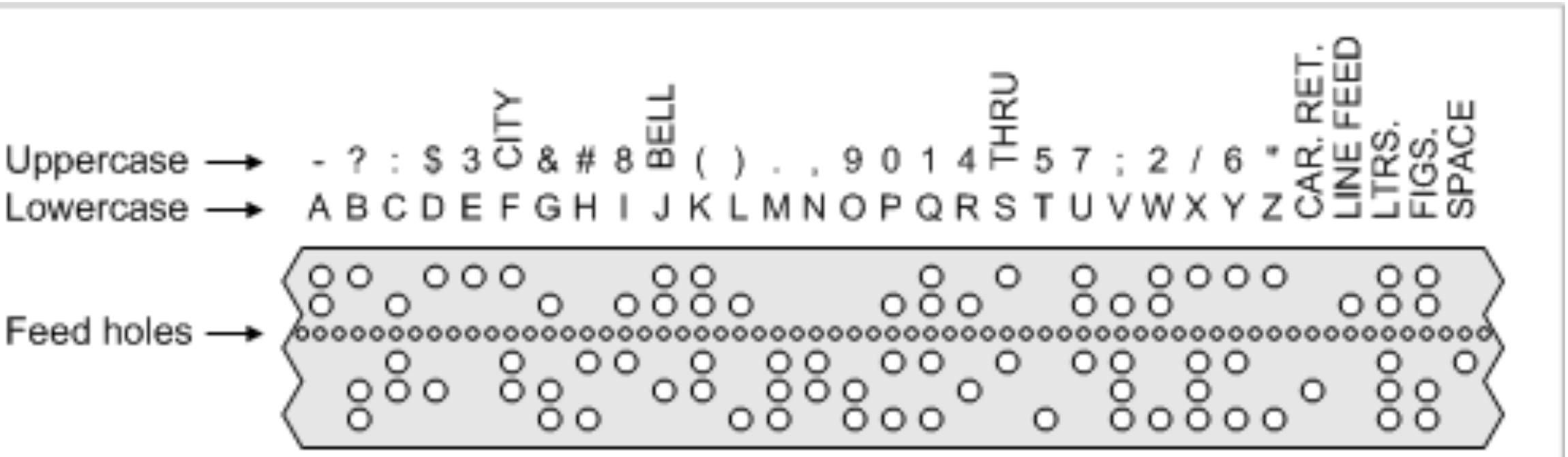
blink.c -> sos.c

Teletype

http://www.smecc.org/police_-__fire_-_civil_defense_communications.htm

Baudot Code

https://savzen.wordpress.com/tag/baudot/



Paper tape showing the five-bit Baudot Code

Baud: Number of symbols per second

e.g. 9600 baud = 9600 bits/sec

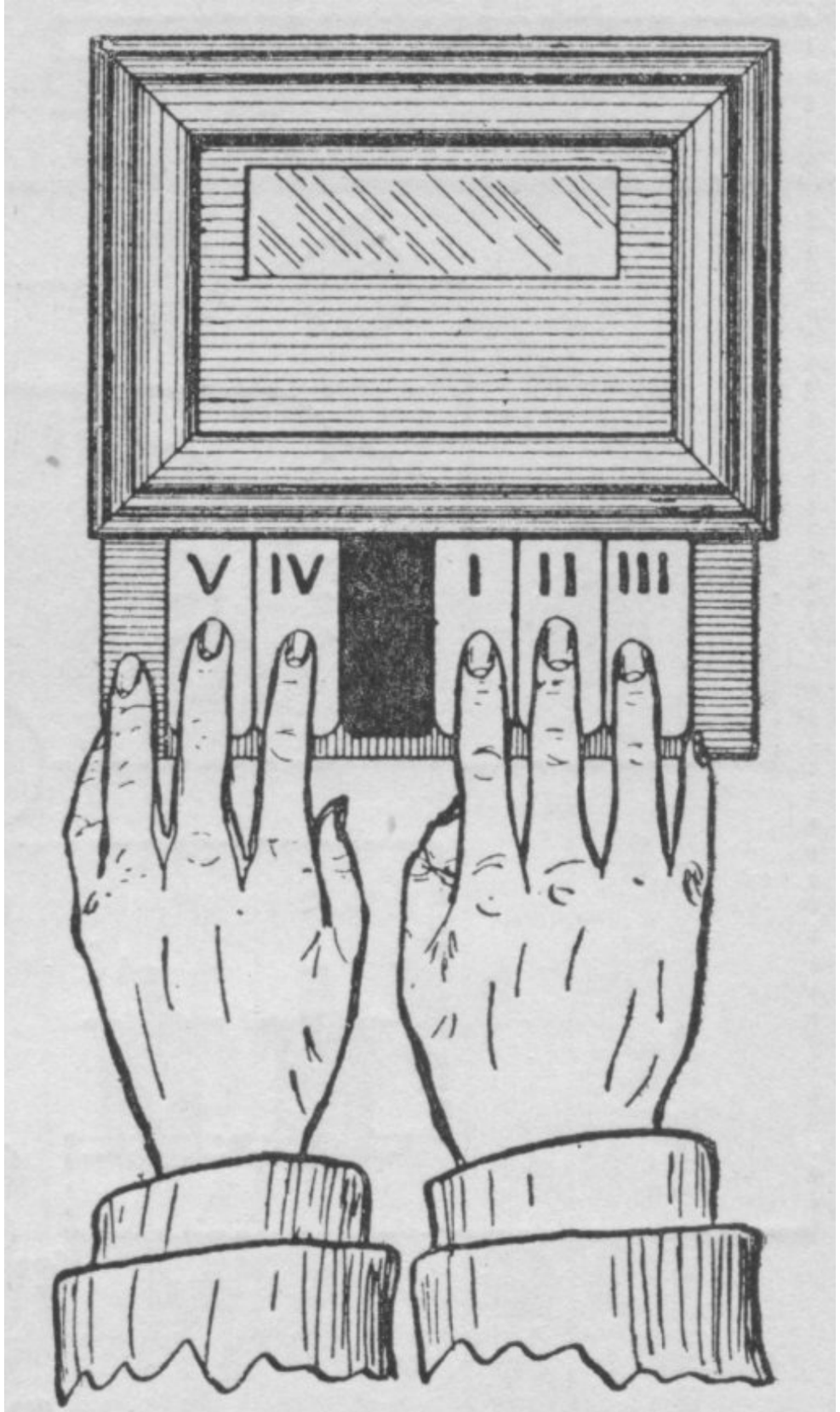


FIG. 17.
BAUDÔT KEYBOARD—
POSITION OF OPERATOR.

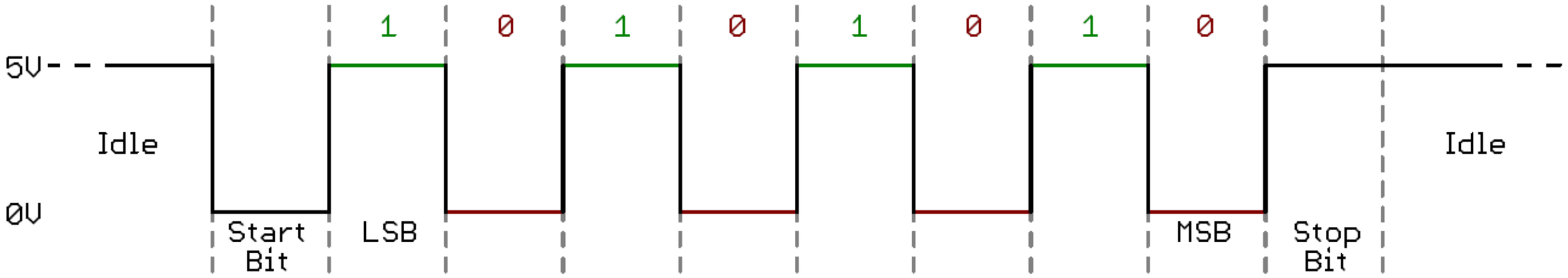
Baudot Code Keyboard

```
% ascii
   2 3 4 5 6 7
  ------------
0:   0 @ P ' p
1: ! 1 A Q a q
2: " 2 B R b r
3: # 3 C S c s
4: $ 4 D T d t
5: % 5 E U e u
6: & 6 F V f v
7: ' 7 G W g w
8: ( 8 H X h x
9: ) 9 I Y i y
A: * : J Z j z
B: + ; K [ k {
C: , < L \ l |
D: - = M ] m }
E: . > N ^ n ~
F: / ? O _ o DEL
```

"cs107e" =

\0
64
37
30
31
73
63

Asynchronous Serial Communication



1 start bit (0), 8 data bits (lsb-first), 1 stop bit (1)

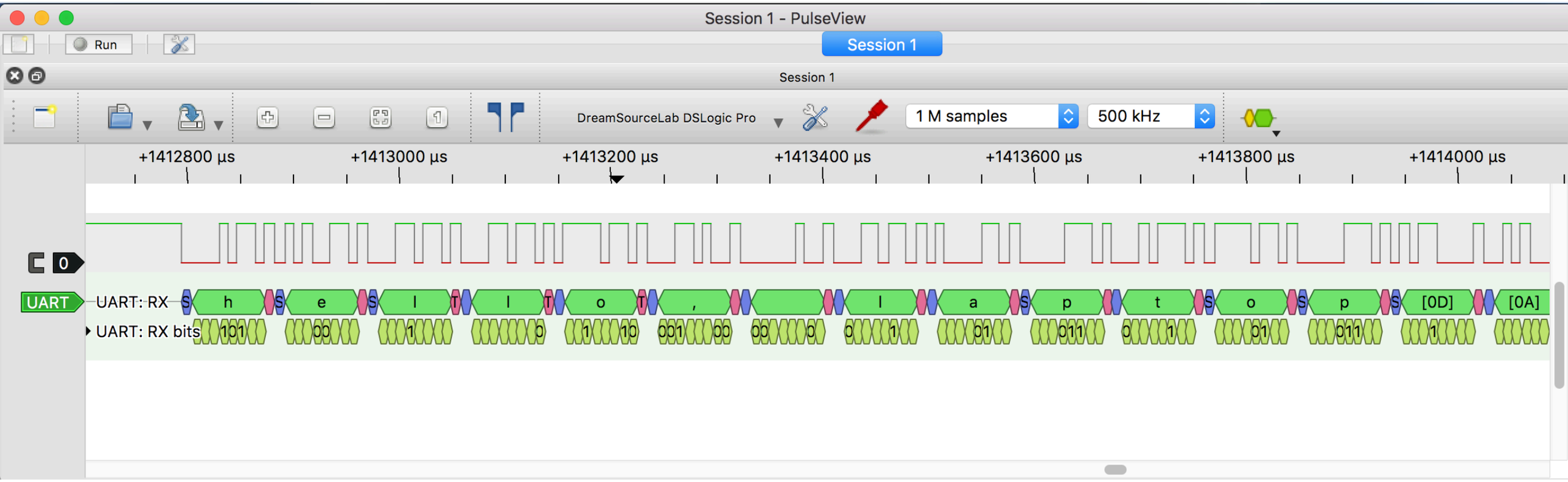
9600 baud = 9600 bits/sec

(1000000 usecs)/9600 ~ 104 usec/bit

https://learn.sparkfun.com/tutorials/serial-communication

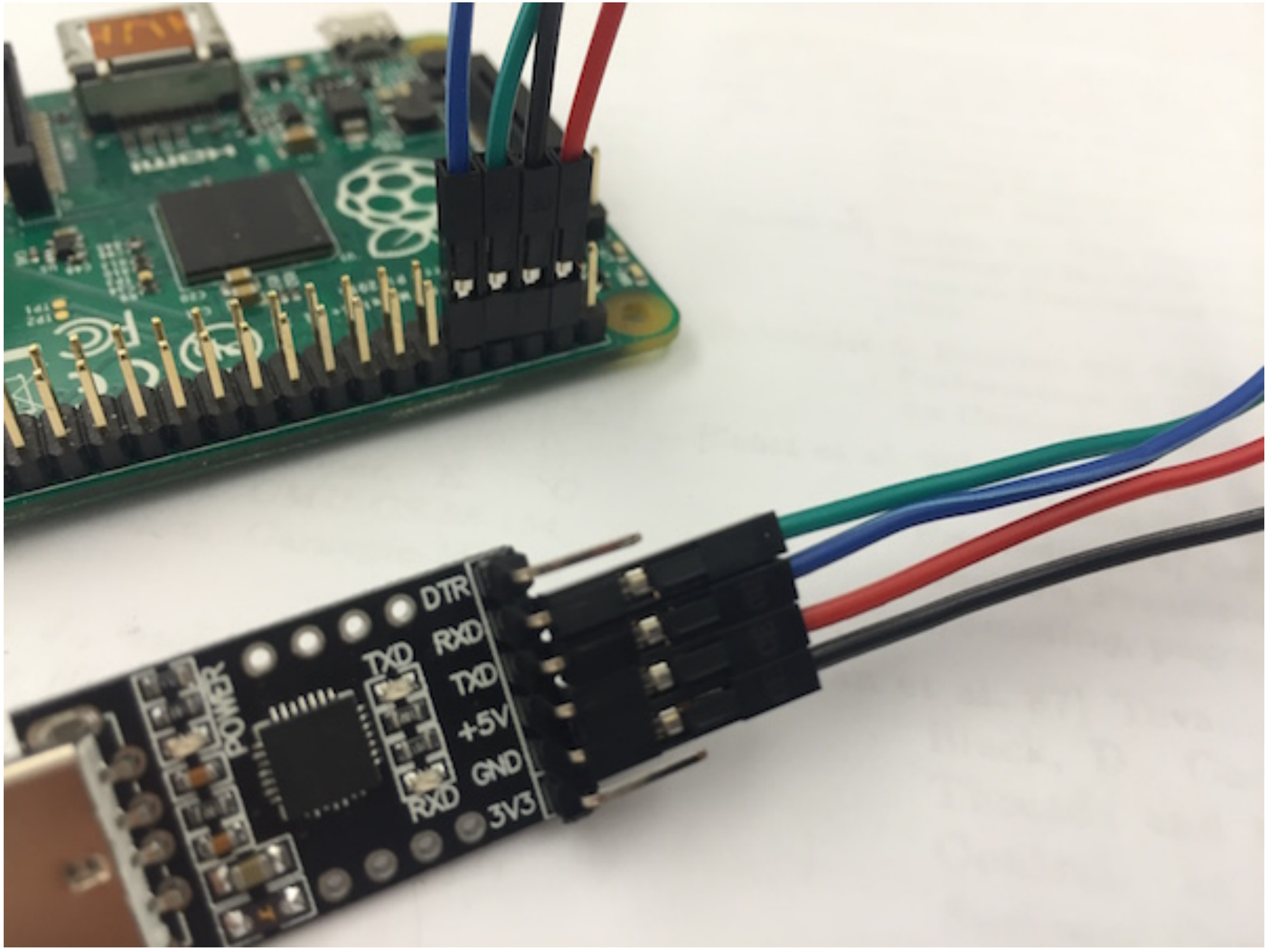
sos.c -> serial.c

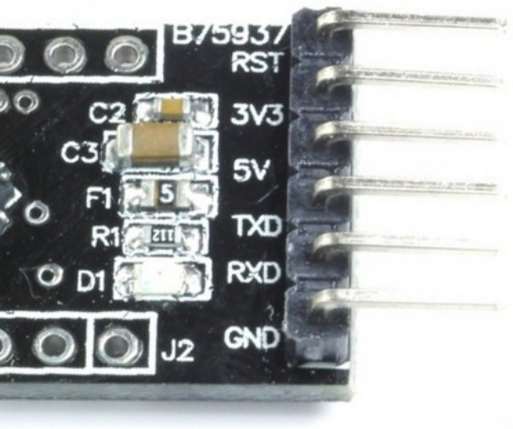
Logic Analyzer!

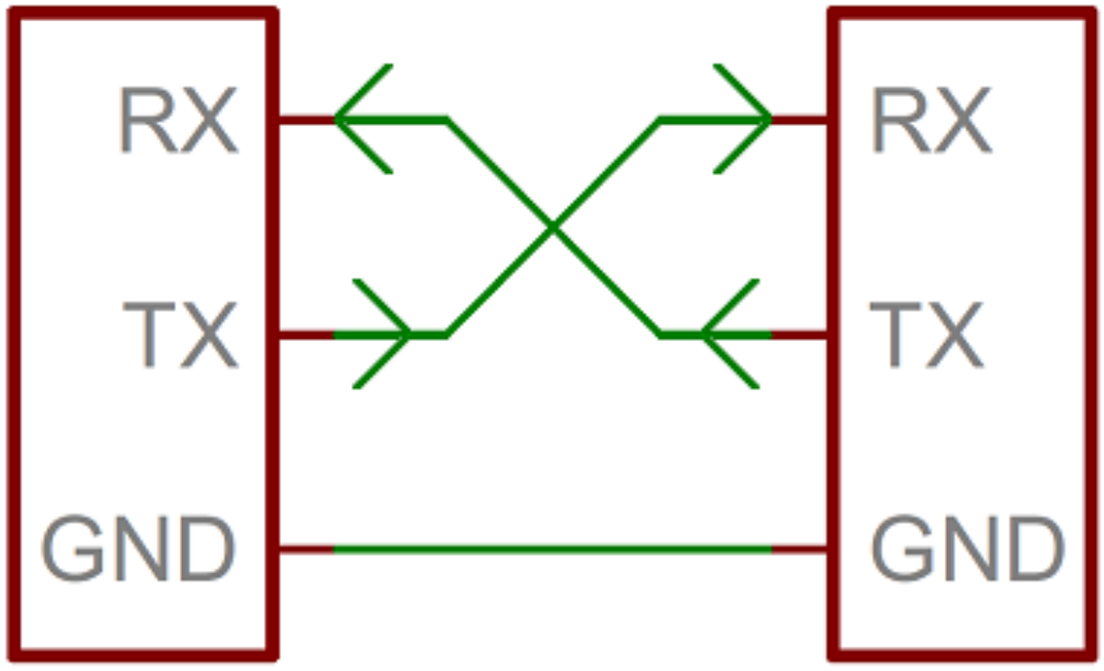


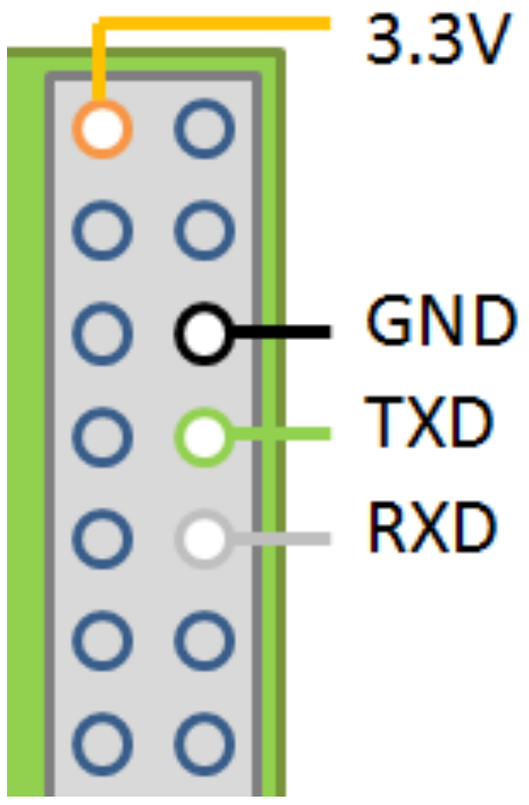
```
// hot wire TX

// device = tty (teletype)
// baud rate = 9600

% screen /dev/tty.SLAB_USBtoUART 9600
```

CTRL-A K - to exit


DTR
RXD
TXD
TXD
+5V
GND
3V3
RXD
POWER








```
% screen /dev/tty.SLAB_USBtoUART 115200
```

uart.c

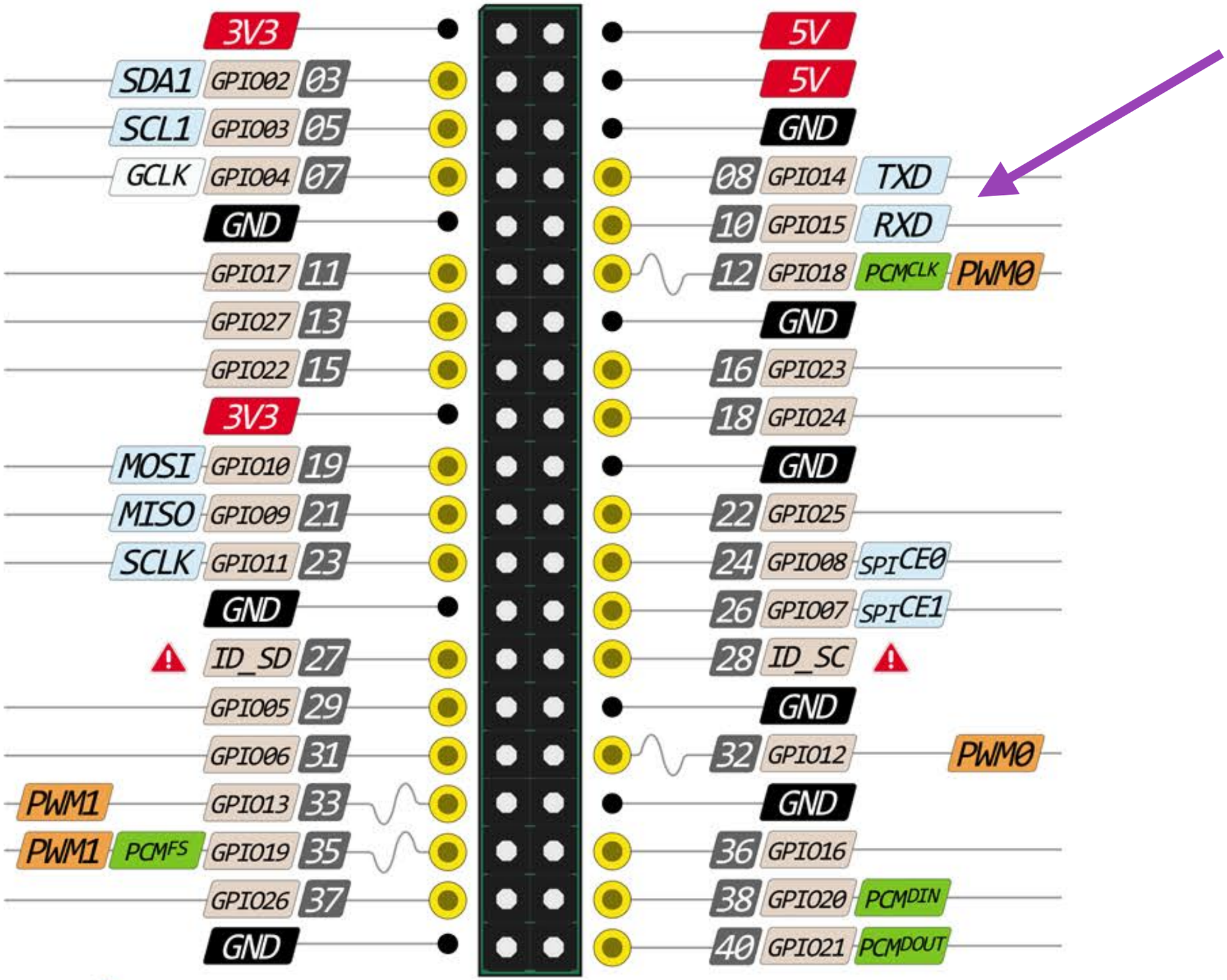
Universal Asynchronous Receiver-Transmitter

GPIO ALT Function

BCM2835 has 54 general-purpose I/O pins. Every pin can be input, output, or one of 6 special functions (ALT0-ALT5), specific to each pin.

PIN	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5
GPIO14	TXD0	SD6				TXD1
GPIO15	RXD0	SD7				RXD1

GPIO Alternate Functions

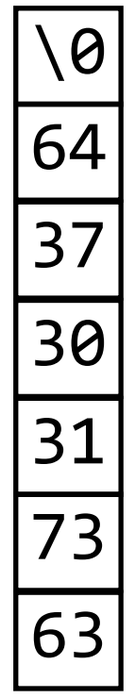


```
// BCM2835-ARM-Peripherals.pdf
// Sec 2: Mini-UART, SPI0, SPI1, pp 8-19
struct UART {
    unsigned data; // I/O Data
    unsigned ier;  // Interrupt enable
    unsigned iir;  // Interrupt identify/fifo
    unsigned lcr;  // line control register
    unsigned mcr;  // modem control register
    unsigned lsr;  // line status
    unsigned msr;  // modem status
    unsigned scratch;
    unsigned cntl; // control register
    unsigned stat; // status register
    unsigned baud; // baud rate register
} ;
```

echo.c

loop back test

C Strings



```
char arr[] = ['c','s','1','0','7','e','\0'];
//char arr[] = "cs107e";
char *ptr = "cs107e";

char ch;
ch = arr[1]; // ok?
ch = ptr[1];

arr = ptr;
ptr = arr;

char **ptrptr;

ptrptr = &arr;
ptrptr = &ptr;
```

String Functions in string.h

```
strcat(s1,s2)       Concatenate s2 to s1
strncat(s1,s2,n)    Concatenate at most n characters of s2  to s1

strcpy(s1,s2)       Copy s2 to s1; Note the direction of the copy!

strncpy(s1,s2,n)    Copy first n characters of s2 to s1
strlen(s)           Return length of string s, not counting ‘\0'

strcmp(s1,s2)       Compare s1 with s2; Return integer less than  zero,
                    equal to zero, or greater than zero
strncmp(s1,s2,n)    Compare only the first n characters of s1 and s2

strchr(s,c)         Return a pointer to first occurrence of character c in
                    string s; return NULL if not found
strrchr(s,c)        Return a pointer to last occurrence of character c in
                    string s; return NULL if not found

strstr(s1,s2)       Return a pointer to the first occurrence of string s1  in
                    string s2; return NULL if not found
strstr(s1,s2)       Return a pointer to the first occurence of string s1  in
                    string s2; return zero if not found
```

```
size_t strlen(const char *str)
{
  for (const char *s = str; *s; ++s)
      ;
  return (s - str);
}
```

```
// strlen("a")?
// strlen(NULL)?
// strlen('a')?
```

```
// Assignment 3

printf(const char *format, …);

printf("%d, %d", 1, 2);
printf("%d, %d, %d", 1, 2, 3);
printf("%d, %d, %d", 1, 2);
```

```
// Read about #include <stdarg.h>
// in the assignment writeup to
// to learn to use functions with
// variable numbers of arguments
```