100ko / min data transmission for Sharp MZ

We need 3 signals on the Sharp MZ:

- WDATA	Adress $E002	Bit 1	PC1	Bit OUT
- MOTOR	Adress $E002	Bit 4	PC4	Bit IN
- RDATA	Adress $E002	Bit 5	PC5	Bit IN

Initialization: We set WDATA = 0, MOTOR = 0 and RDATA = 0 (Point A on the graph).

Fast transmission sequence:

1 - The Sharp MZ requests the positioning of a data: WDATA goes from 0 to 1 (B).

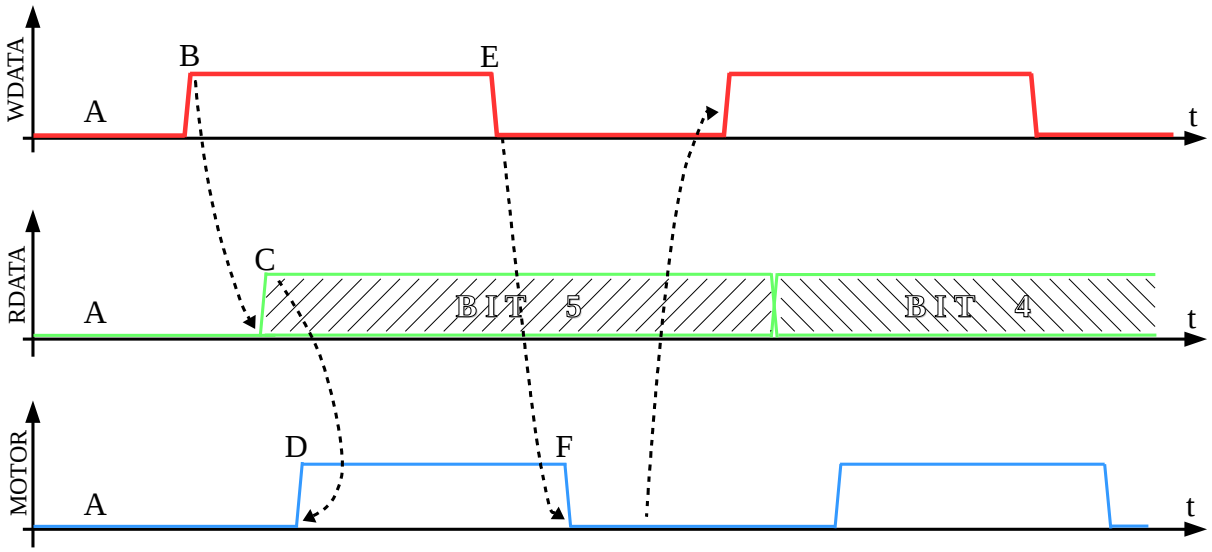
2 - The arduino waits for this signal and sets a bit to 0 or 1 using the RDATA line (C).

3 - The arduino informs the Sharp MZ that the bit is set and ready to read: MOTOR goes from 0 to 1 (D).

4 - The Sharp MZ reads the bit contained in RDATA, then informs the end of reading by setting WDATA to 0 (E).

5 - The arduino acknowledges this signal by setting MOTOR to 0 (F).

The cycle can start again for a new bit.



In order to have the largest possible transfer rate, I use the following methods:

- The bits are transferred in the order of 5 then 4 then 3 then 2 then 1 then 0 then 7 and 6. The Sharp MZ, at each reception, performs a bit rotation with the instruction RLCA, which returns the first bit received at position 5 of the byte (corresponding to PC5 of the RDATA line). So we do not need to do more bit rotation to recover the transmitted byte.

- The transfer program is located in the comments of the conventionally called MZF program by the Sharp MZ monitor because it is only 53 bytes long, and the initial loading is greatly accelerated:

  On Sharp MZ-700, we have for the header:
  2s waiting: 2000000
  100 bits of GAP: 100 * 504
  40 bits at 1: 40 * 958
  40 bits at 0: 40 * 504
  2 bits at 1: 2 * 958
  Header: Maximum 80 bytes of $ FF with 1 synchronization bit: 9 * 80 * 958
  CheckSum: Maximum 2 bytes of $ FF with 1 synchronization bit: 2 * 9 * 958
  1 end bit: 1 * 958

  ie 2818758 µs so 2.82s max

  After loading, the program is in $ 1108, so in the comments.
  In order for the program to run automatically by the monitor, we have to load 3 bytes in memory, mnemonic, JP $ 1108 or C3 08 11.
  For the program to be able to run automatically by the monitor, we have to load 3 bytes in memory, mnemonic, JP $ 1108 or C3 08 11 because in the monitor 1Z-013A we have:

```
0126    2A 06 11        LD HL,(EXADR)
0129    7C              LD A,H
012A    FE 12           CP 12H
012C    38 E1           JR C,LOAD-2
```

  Any program whose execution address is less than 12xxH will not run.

  On Sharp MZ-700, we have for the program:

  2s waiting: 2000000
  100 bits of GAP: 100 * 504 = 50400
  20 bits at 1: 20 * 958 = 19160
  20 bits at 0: 20 * 504 = 10080
  2 bits at 1: 2 * 958 = 1916
  3 bytes (C3 08 11 or 11000011 1 00001000 1 00010001 1): (7 + 3) * 958 + 17 * 504 = 18148
  CheckSum (01110000 1 00000000 1): (3 + 2) * 958 + 13 * 504 = 11342
  1 end bit: 1 * 958 = 958

  ie 2112004 µs so 2.11 s

  Total to transfer the small program and execute it: 2818758+ 2112004 = 4930762 µs so 4.93 s

  Then we go into fast transfer mode.

Example of transfer rate:
EUGEA or SIDEROLL-F programs of 44544 bytes
Transfer in 20.64s
In total, 20.64 + 4.93 = 25.6 s
On 60-4.93 = 55.07 s we are on a fast transfer or 118848 bytes per real min.

ANNEXES

Sharp MZ : Program stored in $ 1108

```
ORG $1108

01 SIZE_H SIZE_L        LD BC, SIZE
21 ADR_H  ADR_L         LD HL, ADRESS

3E 02                   LD A, $02           INIT
32 02 E0                LD ($E002), A       DWRITE = 0
            Boucle1:
E5                      PUSH HL
21 02 E0                LD HL, $E002
11 08 00                LD DE, $0008

            Boucle2:
7E                      LD A, (HL)          WAIT /MOTOR = 0
E6 10                   AND $10
20 FB                   JR NZ, Boucle2

AF                      XOR A
77                      LD (HL), A          DWRITE =  1

            Boucle3:
7E                      LD A, (HL)          WAIT /MOTOR = 1
E6 10                   AND $10
28 FB                   JR Z, Boucle3

7E                      LD A, (HL)          READ BIT
E6 20                   AND $20
B2                      OR D
07                      RLCA
57                      LD D, A             AND STORE

3E 02                   LD A, $02           DWRITE = 0
77                      LD (HL), A

1D                      DEC E
20 E8                   JR NZ, Boucle2

E1                      POP HL
72                      LD (HL), D          STORE DATA TO RAM

0B                      DEC BC              SIZE--
79                      LD A, C             TEST IF BC=0
B0                      OR B
23                      INC HL              ADRESS++

20 D9                   JR NZ, Boucle1      NOT END ? REPEAT !

C3 EX_H  EX_L           JP EXECUTION        PROGRAM EXECUTION
```

Arduino 2560 + PCB + RepRapDiscount Full Graphic Smart Controller

Extract from the program

```
for (i = 0 ; i < mzf_taille ; i++)
 {
  donnee = (uint8_t)(entree.read () & 0xFF) ;
  for (j = 0 ; j < 8 ; j++)
   {
    // Attente demande donnee
    while (digitalRead (MZ_CASSETTE_WRITE) == 0) { }

    // Positionne le bit
    digitalWrite (MZ_CASSETTE_READ, ((donnee & ordre [j]) != 0) ? LOW : HIGH) ;

    // Informe le positionnement du bit
    digitalWrite (MZ_CASSETTE_SENSE, LOW) ; // signal /SENSE a 0 (Lecteur disponible SENSE=1)

    // Attente de la lecture de la donnee
    while (digitalRead (MZ_CASSETTE_WRITE) == 1) { }

    // Reinitialisation
    digitalWrite (MZ_CASSETTE_SENSE, HIGH) ; // signal /SENSE a 1 (Lecteur non disponible SENSE=0)
   }
 }
```