MSX Article

MARMSX

Joynet

Summary

This article aims at presenting an alternative MSX network connection, called Joynet, built using cheap items and using the MSX joystick port.

1- Introduction

The computer communication brings many benefits for us. It is possible to exchange files, share programs, play a network game with another person etc.

The MSX has some communication devices like RS-232 interface and also the modern Ethernet network boards [1][2]. Thus, a cheaper alternative came up from an MSX Internet discussion list [3][4].

But that idea was not an innovation, once Konami had already developed such thing for the game F1 Spirit 3D. Nevertheless, no communication protocols or more detailed informations were left as a reference to the Joynet.

This article propose is to gather all information on the Joynet, showing how to build a MSX-MSX and a MSX-PC cable and how to use the MSX joystick port to accomplish the communication between two MSXs or one MSX and the PC.

2- The Joynet cable

There are two different solutions to connect MSX using the Joynet: one cable to connect two MSXs via joystick port and another one to connect a MSX to a PC via MSX joystick port and PC parallel port.

The items used to build a cable are very cheap.

2.1 – The MSX-MSX cable

The following items are necessary to build a MSX-MSX cable [3]:

- 2 female DB-9 connectors.
- 2 boxes for the DB-9 connectors protection.
- 7 telephone-like wires (25 mm diameter) having 1,5 m length.
- Soldering iron to solder.

The 1 to 9 numeration generally comes printed on the DB-9 connector. Figure 1 shows this numeration from the solder side.

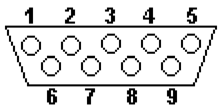

Figure 1. Solder side from DB-9 female connector.

The wire connection between the first and the second DB-9 connector is made using the table 1 as reference. For example, one wire must be soldered on the connector 1 - pin 1 and also on the connector 2 - pin 6.

Connector 1	Connector 2
1	6
2	7
3	8
6	1
7	2
8	3
9	9

Table 1. MSX-MSX connection.

The wire map between the two connectors (table 1) is presented on the figure 2.

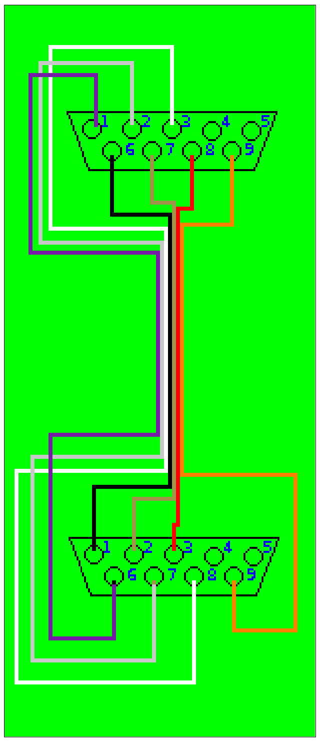

Figure 2. MSX-MSX connection.

2.2 – The MSX-PC cable

The following items are necessary to build a MSX-PC cable [3]:

- 1 female DB-9 connector.
- 1 male DB-25 connector.
- 2 boxes, one for the DB-9 and another for the DB-25 connector protection.
- 7 telephone-like wires (25 mm diameter) having 1,5 m length.
- Soldering iron to solder.

The DB-25 layout is presented on figure 3. This is also the solder side.

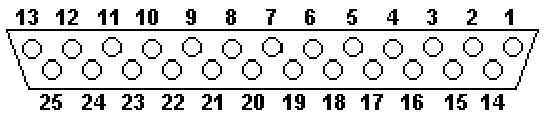

Figure 3. Solder side from the DB-25 connector.

The wire connection between MSX and PC connectors will be accomplished according to the table 2. The wire map is presented on figure 4.

DB-9	DB-25
1	2
2	3
3	4
6	13
7	12
8	10
9	18 ~ 25

Table 2. MSX-PC connection.

The DB-9 - pin 9 must be connected to one of the following DB-25 pins: 18, 19, 20, 21, 22, 23, 24 or 25. These pins corresponds to the PC ground (GND).

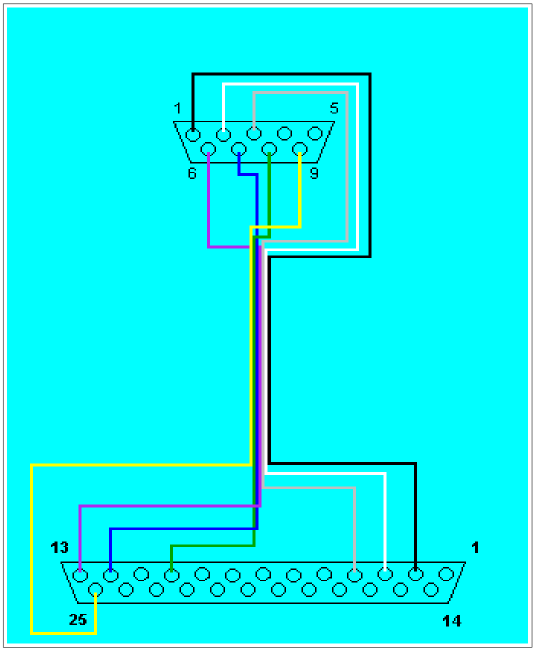

Figure 4. MSX-PC connection.

2.3 – The MSX joystick port

Most MSXs have two joystick ports. These ports are two male DB-9 connectors, were the pin layout (front view) is the same presented on figure 1. Whenever a joystick is connected to any MSX joystick port, the pins work as shown in table 3.

Pin	Function	Signal
1	Up	receive
2	Down	receive
3	Left	receive
4	Right	receive
5	+ 5 Volts CC	-
6	Button 1	both
7	Button 2	both
8	Out	send
9	GND	-

Table 3. MSX DB-9 connector functions.

The joystick is a quite simple circuit, which receives 5 Volts from the MSX pin 5 and have many switches who open/shut the signal to the pins 1, 2, 3, 4, 6 and 7.

The integrated circuit responsible for controlling MSX joystick ports is the PSG AY-3-8910-A. The acronym comes from *Programmable Sound Generator*. According to that, we conclude that the MSX sound processor is also responsible for controlling joystick ports.

The PSG has 15 registers and the communication with MSX is made through ports. Thus, only two registers are used to control joysticks: #14 and #15. The Z-80 access ports to PSG are &HA0, &HA1 e &HA2. The &HA0 is used to select one PSG register, in which the data will be sent using port &HA1 and read using port &HA2.

Register #14 configuration [5]:

Bit	7	6	5	4	3	2	1	0
Description	Cassette in	Keyboard Mode	Button 2	Button 1	Right	Left	Down	Up

Register #15 configuration [5]:

Bit	7	6	5	4	3	2	1	0
Description	Kana LED	Joystick Selection	Pulse 2	Pulse 1	1	1	1	1

The shadow regions indicates that such functions will not be used by the Joynet.

2.4 – Description of the PSG registers #14 and #15

Register #14 uses the PSG A port, which is used to input data. Only 6 bits are used with joysticks, ranging from 0 to 5. All are related to the joystick buttons state. The value 1 means that the respect button/arrow is currently unpressed, or in other words, there is no signal coming back to the PSG, whereas the value 0 means that such button/arrow is being pressed.

Register #15 uses the PSG B port, which is used to output data. Seven bits are used, ranging from 0 to 6. The bit 6 selects the joystick port, where the value 0 selects the joystick #1, while the value 1 selects the joystick #2. The other bits are related to signal emission to the joystick ports through pins 6, 7 and 8, as described as follows.

The two bits (4 and 5) are used to generate short pulse emission to a paddle connected to any MSX joystick port [5]. They will be used to send data to another MSX. According to that, we send the signal using pin 8 in such way that if bit 4 is reset, nothing is sent, while if bit 4 is set, a pulse is sent.

Register #15 bits 0 and 1 indicates which sender pin should emit a continuous signal. The bit 0 corresponds to the pin 6 and the bit 1 corresponds to the pin 7. Both can send data at the same time. The table 4 presents the possible combinations used to emit a signal through the joystick port.

Register 15								Pin 6	Pin 7	Pin 8
7	6	5	4	3	2	1	0			
X	X	0	0	1	1	1	1	0	0	VCC
X	X	0	1	1	1	1	1	0	0	PULSE
X	X	0	0	1	1	1	0	VCC	0	VCC
X	X	0	1	1	1	1	0	VCC	0	PULSE
X	X	0	0	1	1	0	1	0	VCC	VCC
X	X	0	1	1	1	0	1	0	VCC	PULSE
X	X	0	0	1	1	0	0	VCC	VCC	VCC
X	X	0	1	1	1	0	0	VCC	VCC	PULSE

Table 4. Possible signal combinations sent by MSX joystick port.

The value XX001111 is sent back to register #15 after each operation is complete. In order to keep any pin sending data, is necessary to create a loop that changes constantly this value, according to the table 4.

2.5 – Manipulating data on PSG registers

The first step to manipulate data on a PSG register (read/write) is to choose the desired register using the port &HA0. The following Assembly code shows how to.

```
LD A,&H0F      ; Select register #15
OUT (&HA0),A
```

Or

```
LD A,&H0E      ; Select register #14
OUT (&HA0),A
```

Always when a different PSG register has to be used, we must use the port &HA0 to select it. The next step is to read or write on a register. To write use port &HA1 and to read use port &HA2.

Before we start to change registers, we must keep in mind that the modification of one bit value from the register must preserve the other bits. For such thing, we must use AND or OR logical operators.

The AND operator is used to reset some bits and preserve the other. In the other hand, the OR operator is used to set some bits and preserve the other. On the mask used with AND operator, the value 0 changes the bit, while the value 1 keeps the bit value. On the mask used with OR operator, the value 1 changes the bit, while the value 0 keeps the bit value.

For example, suppose that we want to reset the bit 3 from the register #15. The first thing to do is to read the register content. Then, apply the E operation using the proper mask. Finally, send the new value to the register #15.

```
REG_15  = &B00001111
mask    = &B11110111
-------------------- AND
REG_15  = &B00000111
```

And how to set the bit 3?

```
REG_15  = &B00000011
mask    = &B00001000
-------------------- OR
REG_15  = &B00001011
```

This care with bit values is necessary to modify the registers. The bits that will not be change must be preserved from changing. In order to achieve that, use the correct operator and mask.

The following Assembly program is a more complete example, where selects the joystick 1 and reads the buttons/arrows state on Z-80 register A.

```
LD A,&H0F
OUT (&HA0),A  ; Select register #15
IN A,(&HA2)   ; Read register #15 and store in A
AND &HBF      ; Do A AND x0xxxxxx operation
OUT (&HA1),A  ; Write on register #15
              ; Here were selected the joystick 1
LD A,&H0E
OUT (&HA0),A  ; Select register #14
IN A,(&HA2)   ; Register A reads joystick 1 data
```

2.6 – Testing the MSX-MSX cable

The following test is quite simple but necessary to check periodically if the cable is connected to another MSX. Once the sender side pin 8 sends a signal to the receiver side pin 3, we check the bit 2 state on register #14 from the receiver side. While running this test, do not send pulses using pin 8.

The Assembly program that checks the connection is described as follows. The answer is written at MSX RAM address &HC110.

```
       LD A,&H0E       ; Select register #14
       OUT (&HA0),A
       LD A,1
       LD (&HC110),A   ; Set flag as default at &HC110
CHECK: IN A,(&HA2)
       BIT 2,A         ; Check signal on bit 2 (left)
       JR Z,END        ; If set, end
       XOR A
       LD (&HC110),A   ; Else, it is off
END:   RET
```

3 - Credits and References

This article was written originally in Portuguese and translated into English by Marcelo Teixeira Silveira, Systems and Computer Engineer graduated at UERJ.

Written in December 2003.
Reviewed on June 2017.
E-mail: flamar98@hotmail.com
Homepage: http://marmsx.msxall.com

References:

[1] – ObsoNET, Daniel Berdugo at
http://www.konamiman.com/msx/msx-e.html#obsonet
[2] – MSX Ethernet Cartridge, Tecnobytes at
http://www.tecnobytes.com.br/2015/09/msx-ethernet-cartridge-available.html
[3] – Werner Kai at http://www.msxpro.com/mirror/werner/joynetpo.htm
[4] – International MSX Mailing List, offline. Corresponding one:
http://www.msx.org
[5] – The MSX Red Book, Avalon Software, ed. McGraw Hill.