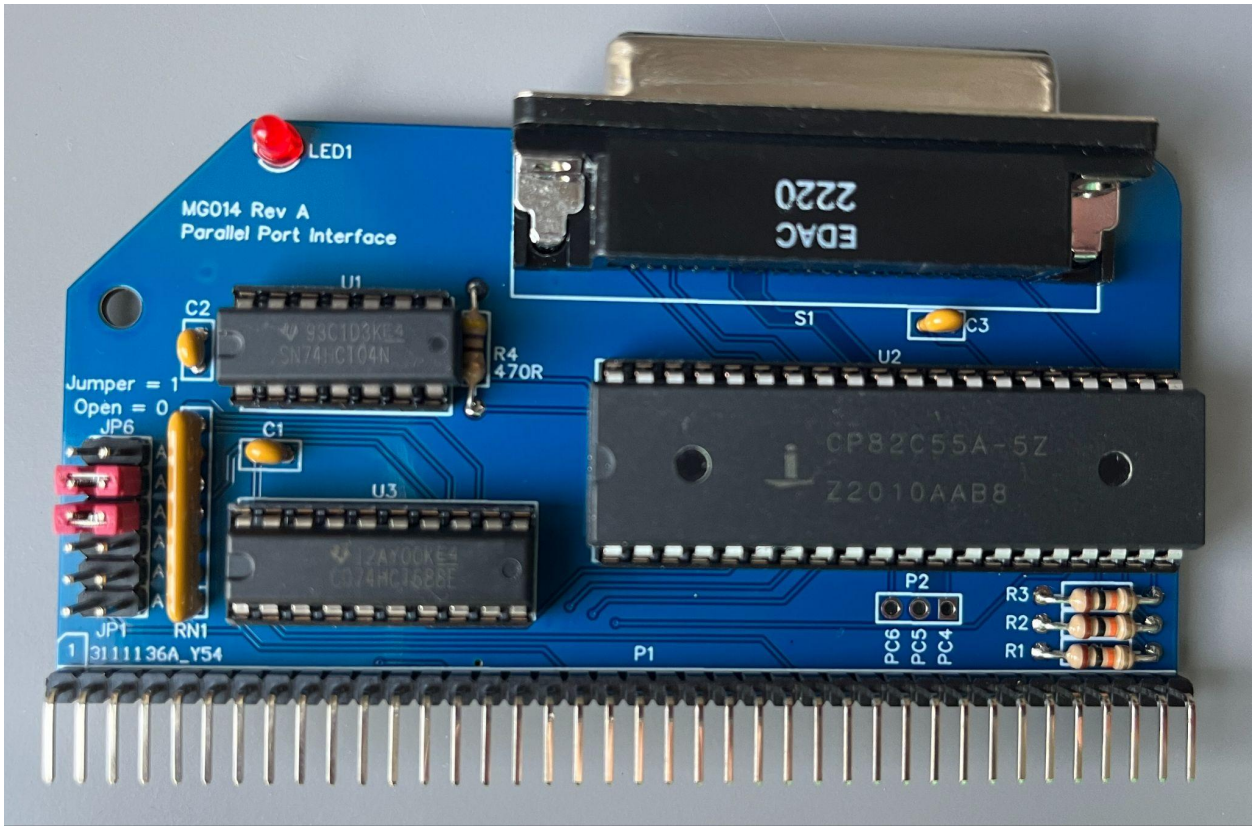
MG014 Parallel Port Interface

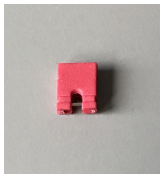
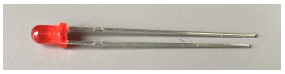
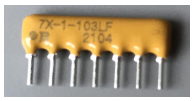
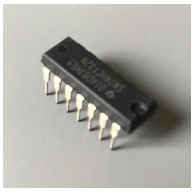
What is it?

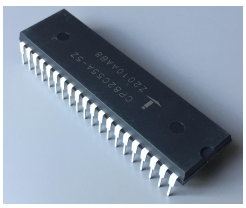
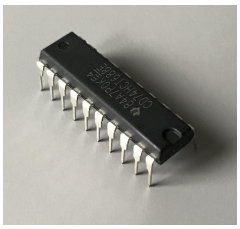
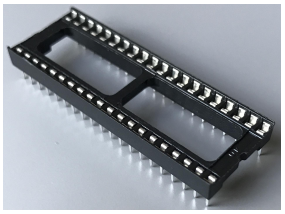
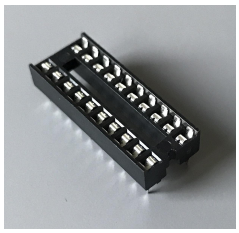
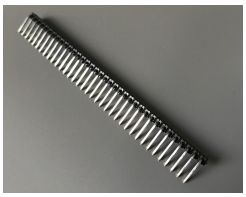
MG014 is a Parallel Port Interface designed for RC2014. It uses the 82C55 Programmable Peripheral Interface, which has 24 I/O pins, which can be flexibly configured as input or output. In the case of MG014, 12 pins are used for output, 5 used for input and 7 are unused.

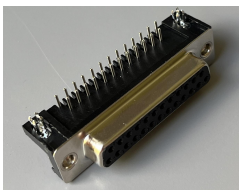
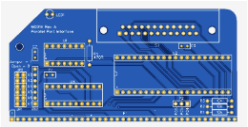
MG014 can be addressed using standard BASIC input and output commands, with one address being used for a control word, and three more for three groups (A, B and C) of I/O pins.



What's in the kit?

Name	Quantity	Description	Picture	Present?
C1-3	3	Capacitor, ceramic, 100 nF		❜
JP1-6	1	Header, male, 2 x 6 pin, straight		❜
JP1-6 Shunt	6	Jumper shunt		❜
LED 1	1	LED, Red, Through Hole, T-1 (3mm), 20 mA		❜
R1 - 3	3	10k Resistor, 250mW		❜
R4	1	470R Resistor, 250mW		❜
RN1	1	Fixed Network Resistor, 10 kohm		❜
U1	1	74HCT04		❜

U2	1	82C55A		,
U3	1	74HCT688		,
U1 socket	1	14-pin DIP socket		,
U2 Socket	1	40-pin DIP socket		
U3 socket	1	20-pin DIP socket		,
P1	1	Pin Header, Right Angle		,

S1	1	D-Sub Socket, 25 pins		,
PCB	1	MG014 PCB		,

How do I build it?

There's a good chance you will have some soldering experience, as you're likely to have built an RC2014 or equivalent to plug your MG014 into. If you haven't, I recommend searching for an online tutorial, there are some good ones on YouTube.

Recommended tools include:

- Soldering iron (ideally temperature controlled)
- Multicore solder
- Small snips to cut off leads
- Small pliers
- Desoldering pump and/or braid
- Anti-static wrist strap (or steer clear of materials that cause static and touch a grounded object every now and then).

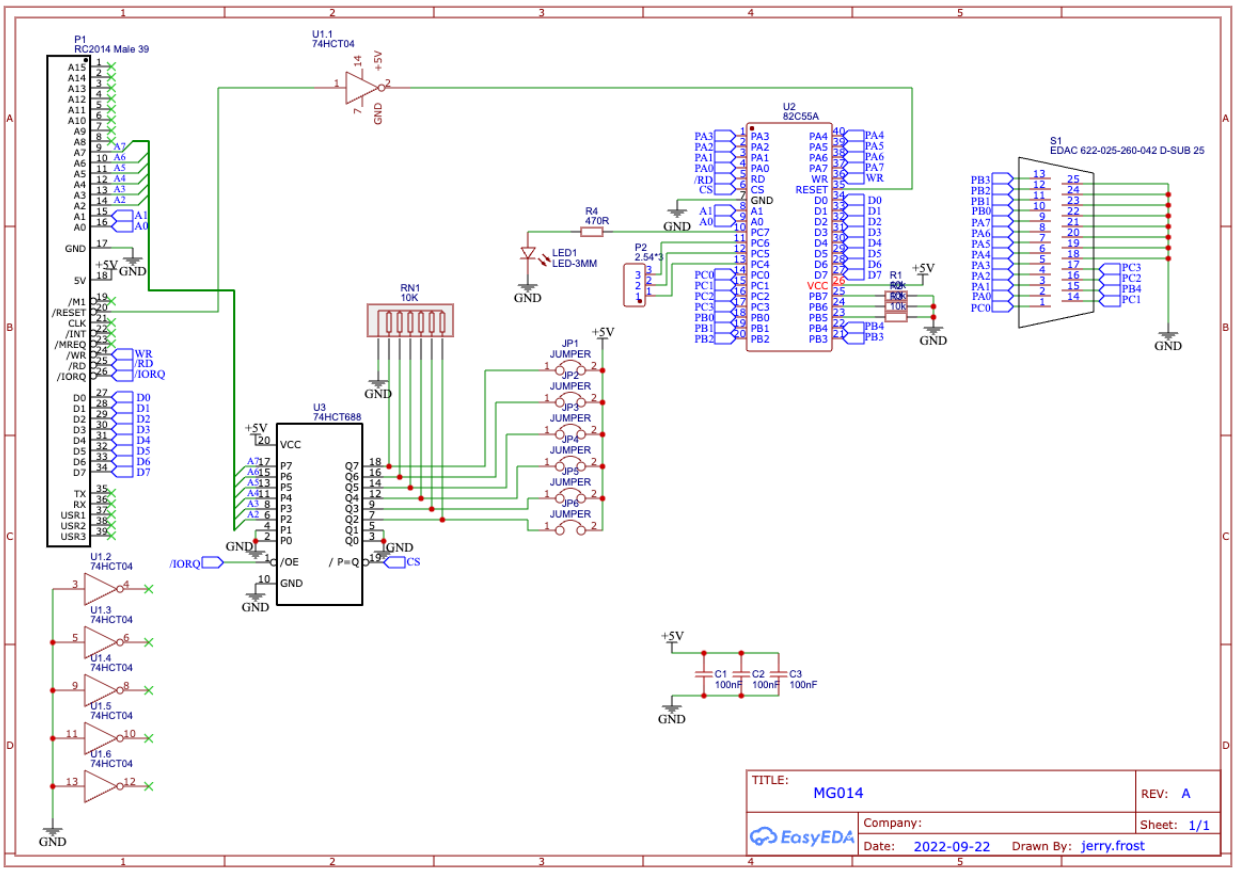
The normal rule of thumb is to solder the lowest height components first, working up:

- P1. This is normally supplied with 40 pins, therefore one needs to be carefully cut off using a sharp knife. Then, solder one joint only, check the alignment, melt solder and correct alignment if required before soldering remaining joints. If supplied with 36 pins, fit at the Pin 1 end of the PCB
- R1-4 Orientation doesn't matter
- C1-3. Orientation doesn't matter
- RN1. Note that the dot on one end should align with the marking on the PCB RN1 graphic
- Sockets for U1-3 (do not fit ICs yet). Similarly to P1, solder two opposite corners, check the socket is flat on the board before continuing. Make sure the notches at the end of the sockets match with the PCB graphics, to reduce the risk of installing the ICs the wrong way round
- LED1. Make sure the flat side is aligned with the graphic on the PCB.
- JP1-6 (actually a single part)
- S1

If you have flux cleaner, clean all joints. Now inspect them carefully for issues (a magnifying glass of some sort can be very helpful, the camera on some phones works quite well).

The final step prior to plugging into the host system and testing is to fit the ICs into their sockets. Their legs will probably need a bit of gentle bending on a table or similar surface, to bring the two rows a little closer to each other. Pay attention to orientation.

How does it work?



JP1-6 are used to set the address of MG014. When the address bus (upper 6 bits) matches this and /IORQ is active, U2 outputs a "/P=Q" signal. This enables U2 for a read or write over the 8 bit data bus, depending on whether /RD or /WR are active.

U1.1 inverts the reset signal (U2 needs an active high) and sets U2 to "all ports input" after a reset.

U2 does all the heavy lifting, it has 24 I/O pins which may be individually programmed in 2 groups of 12 (A plus C Upper, B plus C Lower) and used in 3 major modes of operation (we will use the simplest, Mode 0, throughout these instructions). It's not possible to do justice to all the possibilities here, but they are covered in depth in the Renesas datasheet (and also several online tutorials):

https://www.renesas.com/us/en/document/dst/82c55a-datasheet

For MG014, the groups are used (brought out to P1) as follows:

- A are Data Output (8)
- B are Status Input (5)
- C are Control Output (4)

The pins above fulfill all the standard functions of the parallel port. The spare pins on the 82C55 can all be accessed if for some reason they are needed:

- PB7-5 and PC7 are available on one end of R1-4
- PB4-6 are available at the pads of P2 (a single row header can be soldered in if required but is not supplied).

PC7 is normally connected to LED1 via R4. This can be used as a simple status indicator.

How do I use it?

Parallel ports are (and were) used for many things but by far the most popular is for connection to a compatible printer. The example below will print a basic page of text.

At the risk of stating the obvious, in addition to MG014 and the RC2014, you will need the following items which are not supplied in the kit:

- A printer with a parallel port. You may own one, otherwise please see the section "How can I tell if a printer will work" for some tips on finding one that is likely to work.
- A "Centronics" cable to connect the printer. This should have a DB25 connector on one end to connect to MG014, and a C36M connector to connect to the printer (although check that your printer doesn't have a different connector). These are available from eBay for £5 or less.

When connecting everything up, bear in mind that Centronics cables may be heavy and may try to tilt the MG014 over in its slot. It may be necessary to support the cable or put some insulating packing between MG014 and its adjacent cards.

The example uses the Standard Parallel Port (SPP) mode of operation, which is the most basic one, but likely to be supported by most parallel port printers. (Other modes are discussed in "Parallel Port Modes (In case you were wondering)", for reference).

The first step is to set the address. If you are using the standard RC2014 serial I/O board, then setting A7 to zero (jumper removed) ensures the MG014 will not clash with it. If you have other I/O boards fitted, then you will need to take them into account also.

To use an example:

Jumper	A7	A6	A5	A4	A3	A2
Status	Open	Open	Open	Jumper	Jumper	Open
Value	0	0	0	16	8	0

Summing the values gives a decimal address of 24 (A0 and A1 both zero) to 27 (A0 and A1 both 1). These are used as follows:

- 24 - Port A (Data O/P)
- 25 - Port B (Status I/P)
- 26 - Port C (Control O/P)
- 27 - Control Word to 82C55A

The basics of the Control Word (27) to set A and C to O/P and B to I/P are:

Bit	7	6	5	4	3	2	1	0	Dec.
Port (1=IP, 0=OP)				A	C Upper		B	C Low	
Value	1	0	0	0	0	0	1	0	130

Reading address 27 after a reset will get a result of 155 (all ports IP).
Writing 130 to address 27 will set Ports A through to C ready for use.

The mapping of the bits in Ports A to C is:

Port A

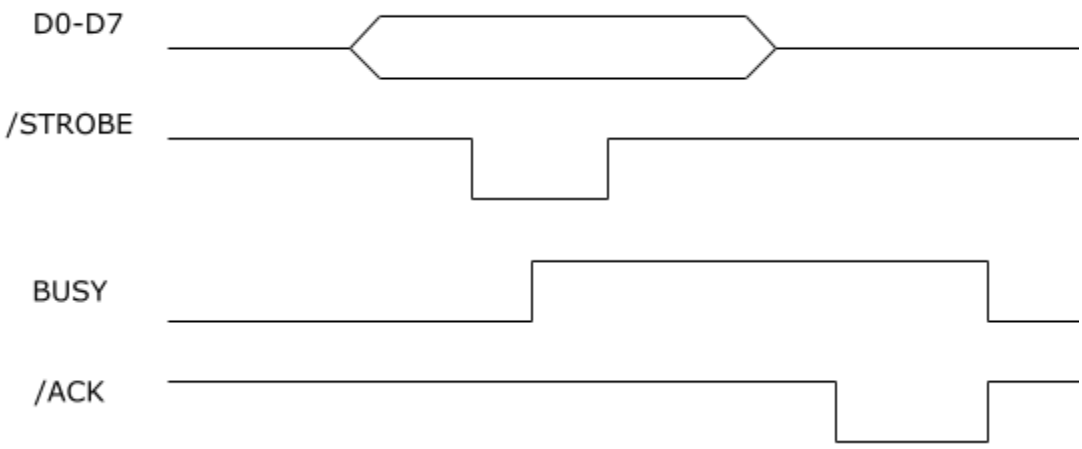
Bit	7	6	5	4	3	2	1	0
	D7	D6	D5	D4	D3	D2	D1	D0

Port B

Bit	7	6	5	4	3	2	1	0
	Not Used (will read "0")			/FAULT	Sel	Paper out	BUSY	/ACK

Port C

Bit	7	6	5	4	3	2	1	0
	MG014 LED	Not Used			/Select in	/INIT	/AUTOFEED	/Strobe



Using the above timing diagram, the basics of use are:

1. The computer checks the peripheral's BUSY line is at logic 0 (not busy)
2. The computer puts the data to be transmitted (ASCII Code) to the peripheral onto D0-D7
3. The computer pulses the strobe line low for a minimum of 500ns[1] to let the peripheral know that valid data is available on D0-D7
4. The peripheral pulses the /ACK line low to confirm it has read the data in.
5. Once /ACK goes high (and BUSY low), the peripheral is ready for the next word.

[1] The exact timings vary between devices. IEEE 1284 (the governing standard for parallel ports) notes several variations regarding exactly how strobe, /ACK and BUSY are timed relative to each other but concludes that they should all work ultimately.

The following example uses BASIC as employed by the RC2014 Classic etc. For users that have an RC2014 Pro or similar, my website (https://jerryfrost1.wixsite.com/tech) provides C source code and executables suitable for CP/M.

The BASIC code is on the following two pages. There's a lot there, but lines 10 to 290 are mainly concerned with getting the text to be printed into an array. Lines 300 to 440 are all that's required to do the actual printing.

There is much that could be done to improve this, perhaps providing "Paper Out" and "Fault" indications. The code as written also doesn't check for overlong lines. The printer I use wraps lines that are too long, others might not.

Please also read the section "ASCII Codes" - it is possible that the BASIC code will need to be modified in areas such as CR/LF usage depending on printer type.

A lot more can be done in C (which is possible with the RC2014 Pro or similar), such as printing the contents of a file (no need to convert text to ASCII numbers).

BASIC Code

```
10 DATA[2] 10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10[3]
20 DATA 27,108,25[4]
30 DATA 32,32,32,32,32,64,64,64,64,64,64,64,64,64,64,64,64,13,10[5]
40 DATA 32,32,32,32,32,64,32,32,32,32,32,32,32,32,32,32,64,13,10
50 DATA 32,32,32,32,32,64,32,32,32,32,32,32,32,32,32,32,64,13,10
60 DATA 32,32,32,32,32,64,32,32,32,32,32,32,32,32,32,32,64,13,10
70 DATA 32,32,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64
75 DATA 13,10[6]
80 DATA 32,32,64,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,64
85 DATA 13,10
90 DATA 32,32,64,32,32,32,32,32,32,32,32,32,32,32,77,71,48,49,52,32,64
95 DATA 13,10
100 DATA 32,32,64,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,64
105 DATA 13,10
110 DATA 32,32,64,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,64
115 DATA 13,10
120 DATA 32,32,64,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,32,64
125 DATA 13,10
130 DATA 32,32,64,32,32,32,64,64,64,64,64,64,64,64,64,32,32,32,64
135 DATA 13,10
140 DATA 32,32,64,64,64,64,64,32,32,32,32,32,32,32,32,64,64,64,64,64
145 DATA 13,10
150 DATA 32,32,32,32,32,32,64,62,95,95,95,95,95,95,95,64,32,32,32,32
155 DATA 13,10
160 DATA 32,32,32,32,32,32,64,32,32,32,32,32,32,32,32,64,32,32,32,32
165 DATA 13,10
170 DATA 32,32,32,32,32,32,64,62,95,95,95,95,95,95,95,64,32,32,32,32
175 DATA 13,10
180 DATA 32,32,32,32,32,32,64,32,32,32,32,32,32,32,32,64,32,32,32,32
```

[2] Lines 10 to 200 contain the data to be sent to the printer. Please see section "ASCII codes" for more details

[3] Line 10 is a sequence of Line Feeds to start the printing half way down the page

[4] Line 20 is a printer ESC code (27 followed by 108) to set the LH margin to 25 characters in (Lines 10 and 20 can be omitted, and the text will print at the top LH side of the page

[5] Line 30 onwards are lines to be printed, characters followed by Carriage Return (13) and Line Feed (10)

[6] I had to split up lines 70 onwards as they were too long to load into RC2014

```
185 DATA 13,10
190 DATA 32,32,32,32,32,32,64,64,64,64,64,64,64,64,64,64,32,32,32,32
195 DATA 13,10
200 DATA 12[7]

250 LET LE = 385[8]
260 DIM XX(LE)
270 FOR Y = 1 TO LE
280 READ XX(Y)
290 NEXT Y

300 OUT 27,130[9]
310 OUT 26,1
320 OUT 26,133[10]
330 IF ((INP(25)) AND 2) = 0 THEN GOTO 360
340 PRINT "PRINTER BUSY"
350 GOTO 330[11]
360 FOR X = 1 TO LE
370 PRINT XX(X)
380 OUT 24, XX(X)[12]
390 OUT 26, 4
400 OUT 26, 5
410 IF ((INP(25)) AND 2) = 0 THEN GOTO 440
420 PRINT "PRINTER BUSY"
430 GOTO 410
440 NEXT X
```

[7] Form Feed, ejects the paper
[8] Lines 250 to 290 load the data above into Array XX(), which is 385 characters long
[9] Sets 82C55 to Ports A and C output, B input
[10] Lines 310 and 320 initialize printer and turn on LED whilst this happens
[11] Lines 330 to 350 loop (printing "Printer Busy") until it isn't. This takes a few seconds whilst it initializes. This code will also cause a permanent "Printer Busy" message if the printer is out of paper or has a fault, which is not necessarily a bad thing.
[12] Lines 360 onwards loop through the data in XX(), printing the code on screen, sending it to printer, pulsing /STROBE low checking it's not busy and starting again

How can I tell if a printer will work?

First, if you have an old parallel port printer, just try it. I didn't possess one, which sent me to eBay (car boot sales, Craigslist etc. are other options). My criteria were:

- Must have a parallel port (obviously)
- I also wanted a USB connection, so that I could try the printer out quickly with a modern PC[13]
- Must be able to buy ink
- Must have the statement "Printer supports direct text printing with the 'us-ascii' charset." on https://openprinting.org/printers

With the above rules satisfied, I found a Canon BJC-85 on eBay. I discovered the BJC-85 and closely related BJC-50 both have detailed service manuals out on the internet, which provided further confirmation that direct text printing via SPP ought to work.

I gambled £25 and bought the BJC-85. It worked, the ink (nowadays) is cheap on eBay, the purchase was a win.

Much of what I have written in these instructions was discovered with this printer and the internet. I don't pretend to be an expert, and there is an awful lot more to parallel port printing than I have learned to date. However, the following points should be noted:

- I suspect that most parallel port printers support direct text printing via SPP. There's no way I can guarantee this is true for any given model, and I recommend potential purchasers do their own research before spending money.
- Printers are likely to differ in areas such as ASCII Control Character support (especially use of CR and LF), ESC code support (noting that you can do a lot of printing without needing ESC codes), exact ASCII Printable Character Set. Experimentation and changes to my example code may be required.

[13] However, many printers old enough to have a parallel port don't have drivers for USB operations with Windows 10, for example. Linux came to the rescue in my case.

Parallel Port Modes (In case you were wondering)

The basic mode of operation of a parallel port is Standard Parallel Port (SPP, also known as compatibility mode). This functions as described above, and is a unidirectional 8 bit method of transferring ASCII codes to the peripheral.

It is possible (if supported by the peripheral) to negotiate to other modes such as Nibble (4 bit transfer from peripheral to host) and ECP Mode (bi-directional transfers). Whilst how to transfer to these modes is documented in IEEE 1284 (and many places online), what I have not been able to find is any description of what the words mean in these modes. It may vary from manufacturer to manufacturer.

SPP works just fine for basic text printing, it will just never be able to support features such as ink level display or graphics printing.

ASCII Codes

There are numerous sources of information regarding ASCII online, but the basics are that “printable” characters are numbered 32 to 126:

Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	Hex	ASCII
32	20		48	30	0	64	40	@	80	50	P	96	60	`	112	70	p
33	21	!	49	31	1	65	41	A	81	51	Q	97	61	a	113	71	q
34	22	"	50	32	2	66	42	B	82	52	R	98	62	b	114	72	r
35	23	#	51	33	3	67	43	C	83	53	S	99	63	c	115	73	s
36	24	$	52	34	4	68	44	D	84	54	T	100	64	d	116	74	t
37	25	%	53	35	5	69	45	E	85	55	U	101	65	e	117	75	u
38	26	&	54	36	6	70	46	F	86	56	V	102	66	f	118	76	v
39	27	'	55	37	7	71	47	G	87	57	W	103	67	g	119	77	w
40	28	(	56	38	8	72	48	H	88	58	X	104	68	h	120	78	x
41	29	)	57	39	9	73	49	I	89	59	Y	105	69	i	121	79	y
42	2A	*	58	3A	:	74	4A	J	90	5A	Z	106	6A	j	122	7A	z
43	2B	+	59	3B	;	75	4B	K	91	5B	[	107	6B	k	123	7B	{
44	2C	,	60	3C	<	76	4C	L	92	5C	\	108	6C	l	124	7C	|
45	2D	-	61	3D	=	77	4D	M	93	5D	]	109	6D	m	125	7D	}
46	2E	.	62	3E	>	78	4E	N	94	5E	^	110	6E	n	126	7E	~
47	2F	/	63	3F	?	79	4F	O	95	5F	_	111	6F	o			□

ASCII numbers 0 to 31 and 127 are control characters:

Dec	Hex	Char	Description		Dec	Hex	Char	Description
0	0	NUL	null		16	10	DLE	data link escape
1	1	SOH	start of heading		17	11	DC1	device control 1
2	2	STX	start of text		18	12	DC2	device control 2
3	3	ETX	end of text		19	13	DC3	device control 3
4	4	EOT	end of transmission		20	14	DC4	device control 4
5	5	ENQ	enquiry		21	15	NAK	negative acknowledgement
6	6	ACK	acknowledge		22	16	SYN	synchronous idle
7	7	BEL	bell		23	17	ETB	end of transmission block
8	8	BS	backspace		24	18	CAN	cancel
9	9	TAB	horizontal tab		25	19	EM	end of medium
10	0A	LF	line feed, new line		26	1A	SUB	substitute
11	0B	VT	vertical tab		27	1B	ESC	escape
12	0C	FF	form feed, new page		28	1C	FS	file separator
13	0D	CR	carriage return		29	1D	GS	group separator
14	0E	SO	shift out		30	1E	RS	record separator
15	0F	SI	shift in		31	1F	US	unit separator
					127	7F	DEL	delete

The use of these control characters varies from printer to printer, but some or all of the following should be supported in some way: LF, CR, ESC, FF[14].

Much has been written about CR and LF, in particular which one of them actually causes the printer to move to a new line[15]. My research has suggested that this varies between printers (I have used a combination of both in my BASIC example). It is also notable that the usage of CR and LF varies across computer operating systems[16].

[14] On the BJC-85 I was surprised to find that BEL actually causes the printer to beep.
[15] The print head on many printers covers several lines of text, it is not unusual for the printer to do nothing until a number of lines have been fed to it.
[16] For Windows, it is CR LF, for UNIX, it is LF, for Mac (up to version 9) it was CR, for MAC OS X and later it is LF.

The use of ESC unlocks further codes, which again seem to vary to some extent between printer types. For example, BJC-85 seems to follow an Epson scheme and 27 108 XX sets the left margin XX columns across. Online documentation is often there to solve these puzzles.

Timings and Compatibility

MG014 is supplied with either 5MHz or 8MHz 82C55 devices, depending on what's available (they are still being manufactured, price is essentially the same for either, but availability can be patchy). I have examined the timing diagrams and confirmed that either device should work equally well.

MG014 does not comply fully with 82C55's need for address/Chip select inputs to be stable 20ns after a write (by about 10ns). I have tested extensively with a Z80 running at up to 20MHz and noticed no issues, and the majority of Z80/82C55 designs and implementations on the web have the same "feature", and are also apparently working fine.

The only timing caution I would note is where users source their own 82C55 from Chinese or similar suppliers on eBay or elsewhere. The risk of these being fakes is high, sadly. If you decide to go this route, then I obviously can not guarantee the results.

Acknowledgements/Legal

MG014 has been designed for RC2014 with reference to the RC2014 Module Template. All pinouts used and the physical outline are in compliance with the RC2014 Module Template.

RC2014 is a trademark of RFC2795 Ltd.

MG014 has been designed for hobbyist use only and is not to be used for safety or business critical applications.