Technical reference Manual

Version 10.0

Stephen Moss

20/12/2021

Contents

Jaguar Console Hardware Release Notes

This document describes the Jaguar Console hardware as far as software development is concerned. It is a companion to the **Jaguar Software Reference Manual – Tom and Jerry**.

General Guidelines for Software

Do not ever write to any of the following registers. The BOOTROM (in a standard retail console) or the STUBULATOR (in a development console) will set them up, Especially the settings in the CLK2, CLK3 and HP registers must be correct to make the hardware work at all and prevent dot crawl in particular. We really mean it: DON'T TOUCH THIS!

MEMCON1	$F00000	**HVS**	$F00036
MEMCON2	$F00002	**HEQ**	$F00054
CLK1	$F10010	**VP**	$F0003E
CLK2	$F10012	**VBB**	$F00040
CLK3	$F10014 (aka CHROMA DIV)	**VBE**	$F00042
HP	$F0002E	**VS**	$F00044
HS	$F00034	**VBE**	$F0004A
HBE	$F00032	**VEE**	$F0004C
HBB	$F00030		

The VMODE register and object processor will be initialised and started after reset by the bootcode. Then the only object in the object list will be a stop object, which will effectively display a blank screen and send the correct video synchronisation signals to the monitor or TV. This also allows the phase locked loop to settle, which takes about a second at start-up. Do not ever turn video off again! (i.e. by writing a zero to VMODE!!)

Specific Bits in Production Series Consoles

Audio is mute after a power on reset. You must turn it on by setting bit 8 of the JOYSTICK register.

Jaguar cartridges usually contain a 128 byte serial EEPROM to be able to save high scores and other user specific information. For information on how to access the EEPROM refer to the installed Drive:\Directory\SOURCE\EEPROM folder of the Jaguar Dev Tools available from Hill Software

EEPROM cartridges currently use bit 0 of JOYSTICK. Do not rely on the readable status of JOYSTICK bit 0 – it is random.

Jaguar Memory Map / Register List

The tables below show the Jaguar hardware register list. For each item in the list, we show the equate as given in the JAGUAR.INC include file (or other appropriate include files), the name of the register as given in the **Jaguar Software Reference Manual**, the address of the register in hexadecimal, and a two letter code for how the register is to be used.

RW = Read/Write WO = Write Only RO = Read Only

Note: Those registers shown in **BOLDFACE** should never be modified by your programs. They are set up for you by the machine at boot-time. They are included here for information purposes only.

System Set-up Registers			
MEMCON1	**Memory Control Register 1**	**F00000**	**RW**
MEMCON2	**Memory Control Register 2**	**F00002**	**RW**
HC	Horizontal Count	F00004	RW
VC	Vertical Count	F00006	RW
LPH	Horizontal Light Pen	F00008	RO
LPV	Vertical Light Pen	F0000A	RO
OB [0-3]	Object Code	F00010-16	RO
OLP	Object List Pointer	F00020	WO
OBF	Object Processor Flag	F00026	WO
VMODE	Video Mode	F00028	WO
BORD1	Border Colour (Red & Green)	F0002A	WO
BORD2	Border Colour (Blue)	F0002C	WO
HP	**Horizontal Period**	**F0002E**	**WO**
HBB	**Horizontal Blanking Begin**	**F00030**	**WO**
HBE	**Horizontal Blanking Begin**	**F00032**	**WO**
HS	**Horizontal Sync**	**F00034**	**WO**
HVS	**Horizontal Vertical Sync**	**F00036**	**WO**
HDB1	Horizontal Display Begin 1	F00038	WO
HDB2	Horizontal Display Begin 2	F0003A	WO
HDE	Horizontal Display End	F0003C	WO
VP	**Vertical Period**	**F0003E**	**WO**
VBB	**Vertical Blanking Begin**	**F00040**	**WO**
VBE	**Vertical Blanking End**	**F00042**	**WO**
VS	**Vertical Sync**	**F00044**	**WO**
VDB	Vertical Display Begin	F00046	WO
VDE	Vertical Display End	F00048	WO
VEB	**Vertical Equalisation Begin**	**F0004A**	**WO**
VEE	**Vertical Equalisation End**	**F0004C**	**WO**
VI	Vertical interrupt	F0004E	WO
PIT [0-1]	Programmable Interrupt Timer	F00050-52	WO
HEQ	**Horizontal Equalisation End**	**F00054**	**WO**
BG	Background Colour	F00058	WO
INT1	CPU Interrupt Control Register	F000E0	RW
INT2	CPU Interrupt Resume Register	F000E2	WO
CLUT	Colour Look-Up Table	F00400-7FE	RW
LBUF	Line Buffer	F00800-1D9E	RW

GPU Registers			
G_FLAGS	GPU Flags Register	F02100	RW
G_MTXC	Matrix Control Register	F02104	WO
G_MTXA	Matrix Address Register	F02108	WO
G_END	Data Organisation Register	F0210C	WO
G_PC	GPU Program Counter	F02110	RW
G_CTRL	GPU Control/Status Register	F02114	RW
G_HIDATA	GPU High Data Register	F02118	RW
G_REMAIN	GPU Division Remainder	F0211C	RO
G_DIVCTRL	GPU Division Control	F0211C	WO

Blitter Registers			
* Must be refreshed after a BLIT ** Must be refreshed if used to store dynamic data (i.e. an inner loop read occurs or GOURD or GOURZ is set) *** Older Version of the **Jaguar Software Reference Manual** (v2.2 & earlier) reversed the order of these descriptions. The equates have not changed, so your source code should be unaffected.			
A1_BASE	A1 Base Register	F02200	WO
A1_FLAGS	A1 Flags Register	F02204	WO
A1_CLIP	A1 Clipping Size	F02208	WO
A1_PIXEL	A1 Pixel Pointer	F0220C	RW*
A1_STEP	A1 Step Value (Integer Part)	F02210	WO
A1_FSTEP	A1 Step Value Fraction (Fractional Part)	F02214	WO
A1_FPIXEL	A1 Pixel Pointer Fraction (Fractional Part)	F02218	RW*
A1_INC	A1 Increment (Integer Part)	F0221C	WO
A1_FINC	A1 Increment (Fractional Part)	F02220	WO
A2_BASE	A2 Base Register	F02224	WO
A2_FLAGS	A2 Flags Register	F02228	WO
A2_MASK	A2 Window Mask	F0222C	WO
A2_PIXEL	A2 Pixel Pointer	F02230	RW*
A2_STEP	A2 Step Value (Integer Part)	F02234	WO
B_CMD	Command/Status Register	F02238	RW*
B_COUNT	Counters Register	F0223C	WO*
B_SRCD	Source Data Register	F02240	WO**
B_DSTD	Destination Data Register	F02248	WO**
B_DSTZ	Destination Z Register	F02250	WO**
B_SRCZ1	Source Z Register 1 (Integer Part)	F02258	WO**
B_SRCZ2	Source Z Register 2 (Fractional Part)	F02260	WO**
B_PATD	Pattern Data Register	F02268	WO**
B_IINC	Intensity Increment	F02270	WO
B_ZINC	Z Increment	F02274	WO
B_STOP	Collision Control	F02278	WO
B_I3	Intensity 3***	F0227C	WO
B_I2	Intensity 2***	F02280	WO
B_I1	Intensity 1***	F02284	WO
B_I0	Intensity 0***	F02288	WO
B_Z3	Z3***	F0228C	WO
B_Z2	Z2***	F02290	WO
B_Z1	Z1***	F02294	WO
B_Z0	Z0***	F02298	WO

Jerry Registers			
CLK1	**Processor clock divider**	**F10010**	**WO**
CLK2	**Video clock divider**	**F10012**	**WO**
CLK3	**Chroma clock divider**	**F10014**	**WO**
JPIT1	Timer 1 Pre-scaler	F10000	WO
JPIT3	Timer 2 Pre-scaler	F10004	WO
JPIT2	Timer 1 Divider	F10002	WO
JPIT4	Timer 2 Divider	F10006	WO
J_INT	Interrupt control Register	F10020	RW
SCLK	Serial Clock Frequency	F1A150	WO
SMODE	Serial Mode	F1A154	WO
LTXD[1]	Left transmit data	F1A148	WO
RTXD[1]	Right transmit data	F1A14C	WO
LRXD[1]	Left receive data	F1A148	RO
RRXD[1]	Right receive data	F1A14C	RO
L_I2S	Left I2S Serial Interface	F1A148	RW
R_I2S	Right I2S Serial Interface	F1A14C	RW
SSTAT[1]	Serial Status	F1A150	RO
ASICLK[1]	Asynchronous Serial Interface Clock	F10034	RW
ASICTRL[1]	Asynchronous Serial Control	F10032	WO
ASISTAT[1]	Asynchronous Serial Status	F10032	RO
ASIDATA[1]	Asynchronous Serial Data	F10039	RW

Joystick Registers			
JOYSTICK	Joystick Register	F14000	RW
JOYBUTS	Button Register	F14002	RW

DSP Registers			
D_Flags	DSP Flags Register	F1A100	RW
D_MTXC	DSP Matrix Control Register	F1A104	WO
D_MTXA	DSP Matrix Address Register	F1A108	WO
D_END	DSP Data Organisation Register	F1A10C	WO
D_PC	DSP Program Counter	F1A110	RW
D_CTRL	DSP Control/Status Register	F1A114	RW
D_MOD	Modulo instruction mask	F1A118	WO
D_REMAIN	Divide unit Remainder	F1A11C	RO
D_DIVCTRL	Divide unit Control	F1A11C	WO
D_MACHI	Multiply & Accumulate High Result Bits	F1A120	RO

[1] The LTXD, RTXD, LRXD, RRXD registers are not listed in the latest version of JAGUAR.INC (Last modified 2/16/95) and therefore presumably should not be used. You could use the L_I2S and R_I2S registers respectively for LTXD, LRXD and RTXD, RRXD as they are included in the latest JAGUAR.INC file and have the same address.
The SSTAT, ASICLK, ASICTRL, ASISTAT & ASIDATA registers are also not listed in the latest versions of JAGUAR.INC and therefore presumably should not be used. It is possible that Atari decided the UART bug was too big a problem to work around and removed them to prevent people from writing networked games. If you want to use these registers, you may have to add them to JAGUAR.INC yourself depending on which version of JAGUAR.INC you have.

Jaguar Video & system Clocks

In the Jaguar Console, the video clock is chosen to allow an inexpensive RF modulator system. This requires slightly different clock speeds for NTSC and PAL systems (but the difference is only about 0.01%). To be cost effective, the GPU/DSP processor clock speed is the same as the video clock speed, and the 68000 is 50% of this clock rate:

	NTSC	PAL
Video Clock GPU/DSP Clock Rate	26.590906 MHz	26.593900 MHz
68000 Clock Rate (50% of Video Clock)	13.295453 MHz	13.296695 MHz

The video system of the Jaguar is programmable within the precision of the supplied video clock. From the video clock, the system produced the pixel (or dot) clock. The ratio between the video and pixel clock is determined by the high order bits of the VMODE register. The possible values for the ratio are shown in the table below, along with numbers of pixel that will fit on the screen overscanned or non-overscanned. The numbers are the same for NTSC and PAL.
For both PAL and NTSC the “safe” video area is about 40µS wide. The area required to guarantee overscan is about 50µS. The table gives the number of pixels that can be displayed within these times for all available pixel clock dividers. Note that these numbers are not ”nice” computer numbers like 320 or 256. Also, note that these are simply rough guidelines to be used in deciding your artwork and object sizes; these numbers should not be used in calculating values for the video hardware register**.**
To properly initialize your program, including video, you must use the standard Jaguar Start-up Code described in the Jaguar Libraries section.

Pixel Divisor value for VMODE register	# of Pixels Non-Overscanned	# of Pixels Overscanned
1	1046	1330
2	532	655
3	355	443
4	266	332
5	213	266
6	177	222
7	152	190
8	133	166

We recommend that ALL software for the Jaguar console overscan both vertically and horizontally so for the rest of this discussion we will restrict ourselves to the OVERSCAN column.

The first row (divisor 1) requires that the object processor be started twice each line and produces a ridiculously high resolution for a TV, so it will be ignored.

A divisor of three gives a non overscanned resolution of about 355. This is a good match for many computer systems and programs designed around 320 pixel wide screens.

A divisor of four gives pixels that are about square. Square pixels are a great advantage for art creation and we recommend their use.

Let’s look at the specific case of an overscanned game using square pixels. This uses a pixel divisor of 4. In both NTSC and PAL this allows for about 332 pixels to be displayed. Choosing a 320 pixel wide bitmap gives us a <4% error. Of these 320 pixels we should only count on the middle 266 being visible on most monitors and/or TV sets. This means that there is a border of about 27 pixels on each side that may be visible, but which should not contain essential game information.

The other pixel clock divisor that is of likely is 5. In this case the number of overscanned pixels is useably close to a blittable width: 256.

To overscan vertically we suggest a screen height of 240 lines for NTSC and 288 lines for PAL. This will allow for both PAL and NTSC users to see a fully overscanned image both vertically and horizontally. The guaranteed visible region within which crucial game information is restricted is 200 lines for NTSC and 240 lines for PAL. Using 200 lines of critical video for both systems is a significant and acceptable simplification.

Video Ports

The information in this section is for informational purposes only. ***Do not attempt to change these timings or unpredictable results will occur!***

There are four versions of the Jaguar console:

Video Standard	Where used
NTSC	USA / Canada
PAL-I	United Kingdom
PAL-B	Germany / other European countries
Peritel/Scart	France

The Jaguar console has an external video connector which supports Composite video, S-Video, and RGB. In addition, there is an RF Modulator output on all versions except the French Peritel/Scart version. The Peritel/Scart version is identical to PAL-B, except that there is no RF modulator. Composite video, S-Video, and RGB are all available on the Peritel version, and have the same timings and characteristics of PAL-B.

The various specification timings are shown below:

RF and Composite

The information in this section is form informational purposes only. ***Do not attempt to change these timings or unpredictable results will occur!***

	Chroma Clock	Subcarrier (MHz)	Sound Carrier (MHz)
PAL-I	4.43361875	591.250	6
PAL-B	4.43361875	591.250	5.5
NTSC Channel 3	3.579545	61.25	4.5
NTSC Channel 4	3.579545	61.25	4.5

Video Timings

The information in this section is form informational purposes only. ***Do not attempt to change these timings or unpredictable results will occur!***

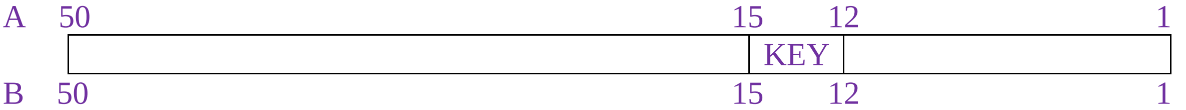
Parameter	PAL	NTSC	
Video master clock	26.593900 MHZ	26.590906 MHz	
Horizontal period	64.0uS	63.5555uS	
Hsync width	4.7uS	4.76uS	
Hback porch	5.7uS	4.45uS	
Hfront porch	1.65us	1.27uS	
Equalisation pulse width	2.35uS	2.54uS	
Vertical sync pulse width	27.3uS	29.26uS	
Vertical lines (interlaced)	625	525	
Vertical Lines (non-interlaced)	624	524	
Vertical sync pulses	5	6	Non-interlaced
Vertical eq pulses before sync	5	6	Non-interlaced
Vertical eq pulses after sync	6	6	Non-interlaced
Vertical front porch	12 lines	12 lines	Non-interlaced
Vertical back porch	17 lines	12 lines	Non-interlaced

Jaguar Console Hardware Ports

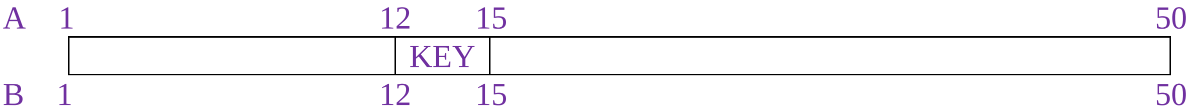
Cartridge / Expansion Port

The Cartridge/Expansion port is a custom 50 pin, double row PCB mounting edge connector.
The far (back) row is row A, and the near row is row B, when looking at the console from the front.

Component side view:



Solder side view:

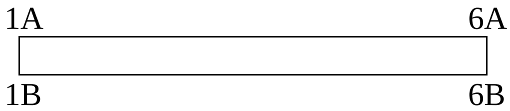


Pin	A	B
1	EA10	GND
2	EA9	GND
3	EA11	EA23
4	EA8	EA22
5	EA12	EA21
6	EA7	EA20
7	EA13	EA19
8	EA6	GND
9	EA14	NC
10	EA5	GND
11	EA15	NC
12	EA4	GND
13	**KEY**	**KEY**
14	**KEY**	**KEY**
15	EA16	EA1
16	EA3	EA0
17	EA17	WAITL
18	EA2	RESETL
19	EA18	EWE0L
20	ROM1	EWE2L
21	GND	ERW
22	ED15	EOE1L
23	ED0	EOE0L
24	ED7	GND
25	ED8	EINT0
26	ED14	EINT1
27	ED1	9V

Pin	A	B
28	ED6	NC
29	ED9	GND
30	ED13	UART1
31	ED2	UART0
32	ED5	GND
33	ED10	RESET1L
34	ED12	CART_IN
35	ED3	CART_OUT
36	ED4	VCC
37	ED11	VCC
38	VCC	PLL
39	ED31	NC
40	ED16	E2DATA
41	ED23	NC
42	ED24	GPIO0
43	ED30	GPIO1
44	ED17	GPIO2
45	ED22	GPIO3
46	ED25	GPIO4
47	ED29	SCK
48	ED18	WS
49	ED21	TXD
50	ED26	RXD
51	ED28	GND
52	ED19	ECPUCLK
53	ED20	GND
54	ED27	GND

DSP Port

The external DSP port is a custom 12 pin, two row edge connector. The top row is row A, the bottom row is row B. Pin 1 is on the left, pin 6 is on the right when looking at the console from the rear:

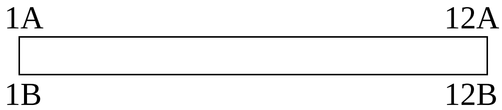


Pin Number	Name	Description
1A	GND	Ground
2A	SCK	Synchronous serial clock
3A	WS	Synchronous serial word strobe
4A	TXD	Synchronous serial transmit data (data out)
5A	RXD	Synchronous serial receive data (data in)
6A	GND	Ground
1B	+5V	+5V, 50mA maximum load
2B	UART_TXD	Asynchronous transmit data (data out)
3B	UART_RXD	Asynchronous receive data (data in)
4B	Reserved	Do not connect
5B	Reserved	Do not connect
6B	GND	Ground

All active signals have 5V TTL levels. The SCK, WS, TXD and RXD signal are also connected to the cartridge expansion connector. They are used on the CD ROM peripheral; therefore, care must be taken to avoid contention (see the audio subsystem section below).

Video Connector

The external video connector is a custom 24 pin, two row edge connector. The top row is row A, the bottom row is row B. Pin 1 is on the left, pin 12 on the right when looking at the console from the rear:



Pin Number	Name	Description
1A	Audio_Left	EIAJ Line Level, left audio
2A	Audio_Gnd	Audio Return (ground)
3A	Reserved	
4A	Video_Gnd	Video Return (ground)
5A	Blue	Blue video, 75 Ohm, 0.7V peak-to-peak
6A	HSync	Horizontal Sync, 75 Ohm, 3.0V peak-to-peak
7A	Green	Green video, 75 Ohm, 0.7V peak-to-peak
8A	Chroma	S-Video Chroma, 75 Ohm, 1.0V peak-to-peak
9A	Reserved	
10A	Reserved	
11A	9V	9V DC, 100mA maximum load
12A	Reserved	
1B	Audio_Right	EIAJ Line Level, right audio
2B	Audio_Gnd	Audio Return (ground)
3B	Video_Gnd	Video Return (ground)
4B	Red	Red video, 75 Ohm, 0.7V peak-to-peak
5B	VSL	Composite Sync, +5V, TTL Levels
6B	Reserved	
7B	Video_Gnd	Video Return (ground)
8B	Luma	S-Video Luma, 75 Ohm, 1.0V peak-to-peak
9B	Reserved	
10B	Video_Gnd	Video Return (ground)
11B	Composite	Composite Video, 75 Ohm, 1.0V peak-to-peak
12B	Reserved	

The Reserved signals should be left unconnected. They may be used in future versions of the Jaguar console, and therefore should be passed through on video adaptors. It is important to terminate the active signals correctly. Do not load the 75 Ohm outputs with more than 75 Ohms.

Multi-Console Games

There are two types of Multi-Console games. The first type uses a special Local-Area-Network of multiple Jaguar consoles connected together via the consoles asynchronous serial port. The second type uses the Jaguar modem to connect two Jaguar consoles via telephone lines.

Jaguar Network

The low-level drivers required for networking multiple Jaguar consoles are currently in development. Contact Jaguar Development Support for further information.

Jaguar Modem

The specification for using the Jaguar modem is described in the document titled **Jaguar Voice Modem**.

Jaguar Controller and Controller Ports

There are two controller ports on the Jaguar console: Controller port 1 (left) and Controller port 2 (Right). Each has the following functions:

- Four bi-direction digital pins
- Six input only digital pins (split into 4 + 2 button)

Note: Early versions of the Jaguar included an 8 bit ADC[2] on the motherboard. This has been deleted – analogue controllers now require their own ADC chip.

Signals and Pin outs

Pin #	Port 1	Port 2	Description
1	J3	J4	Bi-directional signal Used as output to specify to controllers which data to return
2	J2	J5	Bi-directional signal Used as output to specify to controllers which data to return
3	J1	J6	Bi-directional signal Used as output to specify to controllers which data to return
4	J0	J7	Bi-directional signal Used as output to specify to controllers which data to return
5			Reserved
6	B0 / LP	B2	Button input / Light Gun on Port 1
7	+5V DC	+5V DC	+5V, 50mA maximum load
8	n/c	n/c	Pulled up to +5V on 4 player adaptor
9	Gnd	Gnd	Ground
10	B1	B3	Button input
11	J11	J15	Input only signal
12	J10	J14	Input only signal
13	J9	J13	Input only signal
14	J8	J12	Input only signal
15			Reserved

Signal J0 – J15 and B0 –B3 are all TTL level digital inputs and outputs.

Controller Port1 also has a light gun input in addition to the signals mentioned above. A TTL rising edge on the LP signal (pin 6 of Port 1, shared with B0) causes the light pen registers (LPH and LPV) to be latched.

2 Analogue to Digital Converter – a device that converts analogue signals such as a variable voltage level into an 8 bit digital format suitable for processing by a computer.

Register Addressing – Digital Inputs

The table below shows the purpose of the individual bits of the JOYSTICK and JOYBUTS registers. Please note that some bits are used for non controller related purposes.

JOYSTICK	$F14000 15......8 7......0	Read/Write	
Read	fedcba98 7654321q	f-1 q	Signals J15 to J1 Cartridge EEPROM output data
Write	exxxxxxm 76543210	e x m 7 – 4 3 – 0	1 = enable J7 to J0 outputs 0 = disable J7 to J0 outputs don't care audio mute 0 = Audio muted (reset state) 1 = Audio enabled J7 - J4 outputs (Port 2) J3 – J0 outputs (Port 1)
JOYBUTS	$F14002 15......8 7......0	Read Only	
Read	xxxxxxxx rrdv3210	r r d v 3 – 2 1 – 0	don't care reserved reserved 1 = NTSC Video hardware 0 = PAL Video hardware Button Inputs B3 & B2 (Port 2) Button Inputs B1 & B0 (Port 1)

Device Addressing

All controller devices are addressed through the digital lines on the controller ports. Each controller port has 4 bi-directional pins and 6 input pins. We always use the bi-directional pins as outputs. By writing a 4 bit code to these outputs, 16 rows containing 6 bits of data each can be addressed. Each controller is allocated 4 rows of data, so up to four controllers may be connected to each port (via a 4-player adaptor) for a maximum of 8 controllers in total. Controllers may be connected to the Jaguar in two ways:

1. Directly to the Controller port.
2. Via a multi-player adaptor (usually a 4 player adaptor).

Reading a Jaguar Controller

Reading a controller is done in two steps:

1. Write a 4 bit code to the port's output bits which specifies which row of controller data you want to read. Bits 3-0 of the JOYSTICK registers contain the output bits for Port1. Bits 4-7 specify the output bits for port 2. Note that the codes used for port 2 are a mirror image of the codes for port 1 (the bit order is reversed).
   Bit 15 of JOYSTICK must also be set to enable the outputs. Bit 8 is also used to control audio muting, so you have to be careful not to clear this bit accidently or you will disable your program's sound generation.

2. Read back the values contained in the JOYBUTS and JOYSTICK registers. These will contain the 6 data bits returned by each port.

For example, writing a value of $817E to JOYSTICK would allow you read row 0 of the first controller connected to Port 1 and the first controller connected to Port 2. This value breaks down as:

```
$8000 = Enable JOYSTICK outputs J0-J7
$0100 = Enable Audio (bit 8 of JOYSTICK controls audio mute)
$0070 = Setup read of row 0 (code %0111) of controller 0, port 2.
$000E = Setup read of row 0 (code %1110) of controller 0, port 1.
-----
$817E = Value to write to the Joystick register.
```

Below is a table that shows how the six bits of data for each row are returned by the first controller connected to port 1 and the first controller returned on port 2. The meaning of the bits depends on which row is being read and what type of controller is connected (as defined later in the descriptions of each controller type)

Controller Port 1										
Output Pin #					Input Pin #					
1 (J3)	2 (J2)	3 (J1)	4 (J0)		6 (B0)	10 (B1)	14 (J8)	13 (J9)	12 (J10)	11 (J11)
0	1	1	1	Row 3	C3	data	data	data	data	data
1	0	1	1	Row 2	C2	data	data	data	data	data
1	1	0	1	Row 1	C1	data	data	data	data	data
1	1	1	0	Row 0	data*	data	data	data	data	data
Controller Port 2										
Output Pin #					Input Pin #					
1 (J4)	2 (J5)	3 (J6)	4 (J7)		6 (B2)	10 (B3)	14 (J12)	13 (J13)	12 (J14)	11 (J15)
0	1	1	1	Row 3	C3	data	data	data	data	data
1	0	1	1	Row 2	C2	data	data	data	data	data
1	1	0	1	Row 1	C1	data	data	data	data	data
1	1	1	0	Row 0	data*	data	data	data	data	data

* Bit **B0** on Port 1 and bit **B2** on Port 2 are used as a special "Bank 0" flag by bank switching controllers. See **Reading Bank Switching Controllers** for more information.

Identifying Controller Types

The basic type of controller is specified by the **C2** & **C3** bits returned when you read the controller, as shown in the table below. The currently defined controller type identifiers are: [3]

C2	C3	Controller Type
0	0	Reserved
0	1	Bank Switching (analogue Joystick, head-mounted tracker, etc.)
1	0	“Tempest” Rotary
1	1	“Standard” Jaguar Joypad (or nothing connected)

Software must scan all possible controller positions, including those on a 4-player adaptor, to determine which types of controller are currently connected. The software should then offer the user a choice of which compatible controller type to use (where more than one type is attached) or prompt them to attach a supported controller type when no supported controller type is found as applicable.

The identifying connected controller scan must ensure that the Row 0 code for each controller socket tested is valid for at least 100uS. This to allow for the fact that an advanced controller may be attached, as is only necessary during this phase, after which Row 0 durations can be set to the minimum specified for the identified controller type(s). The identifying connected controller scan should be considered a separate form of controller read, its only purpose is to allow the quick identification of an attached controller type within a single four row read.

Advanced controllers use a special bank-switching technique to return more information than the 24 bits of data available from a standard controller. This makes a wide verity of controller types possible, so the specific controller type is identified by certain bits in the last bank of data returned by each controller.

Data Returned from Last Bank (Data Bits B1/B3)				
Row 3	Row 2	Row 1	Row 0	Bank Switching Controller Type
0	0	0	0	Reserved
0	0	0	1	Reserved
0	0	1	0	Reserved
0	0	1	1	Reserved
0	1	0	0	Reserved
0	1	0	1	Reserved
0	1	1	0	Reserved
0	1	1	1	Head-mounted Tracker
1	0	0	0	Reserved
1	0	0	1	Reserved
1	0	1	0	Reserved
1	0	1	1	Reserved
1	1	0	0	Reserved (PS Controller)
1	1	0	1	Keyboard / Mouse
1	1	1	0	6D Controller
1	1	1	1	Analogue Joystick or Driving Controller

See the description of the individual controller types and the section **Reading Bank Switching Controllers** for additional information.

[3] Please note that the specification for identifying controllers was changed on March 31, 1995. The differences are important, but fairly minor from an implementation point of view, and do not affect any existing hardware on the market as of that date.

Standard Jaguar Controller Matrix

Below is a table showing the matrix for the standard Joypad controller which is packed out with every Jaguar console. When plugged directly into the console, the matrix for this controller is as follows:

J4	J5	J6	J7	Port 2	B2	B3	J12	J13	J14	J15
J3	J2	J1	J0	Port 1	B0	B1	J8	J9	J10	J11
0	0	0	0							
0	0	0	1							
0	0	1	0							
0	0	1	1							
0	1	0	0							
0	1	0	1							
0	1	1	0							
0	1	1	1	Row 3	C3	Option	#	9	6	3
1	0	0	0							
1	0	0	1							
1	0	1	0							
1	0	1	1	Row 2	C2	C	0	8	5	2
1	1	0	0							
1	1	0	1	Row 1	C1	B	*	7	4	1
1	1	1	0	Row 0	Pause	A	Up	Down	Left	Right
1	1	1	1							

Reading a zero means the appropriate button is depressed.

Rotary “Tempest” Controller

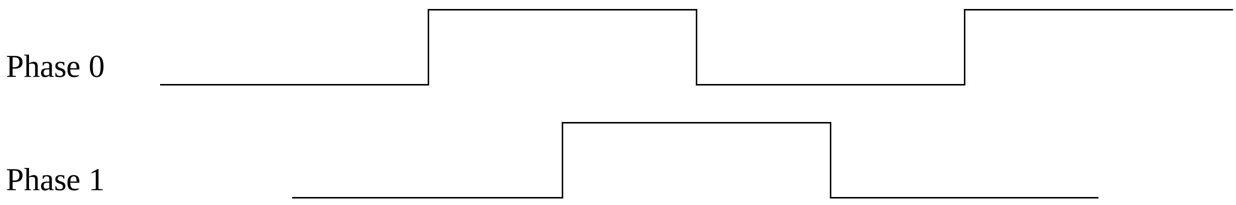
With one or two possible exceptions all existing rotary controllers ere aftermarket products created by Jaguar owners modifying standard Joypad controllers; consequently, they should be read just like a standard controller using Socket 0 row codes which will make the matrix for this controller type as follows:

	B2	B3	J12	J13	J14	J15
	B0	B1	J8	J9	J10	J11
Row 3	0 (C3)	Option	#	9	6	3
Row 2	1 (C2)	C	0	8	5	2
Row 1	1 (C1)	B	*	7	4	1
Row 0	Pause	A			Phase 0	Phase 1

Because these controllers will have no Up and Down function it is recommended that for menu navigation (e.g. Option menu) the following buttons are used:
A = Up B = Select/Change C = Down

This device is similar to the original Tempest arcade controller. It uses a two phase switch, which can be read by software to determine the direction of rotation.

The phase signals (**Phase 0** and **Phase 1**) specify which direction the rotary wheel is turning. The output sequence is a 2 bit grey code that looks like this when the wheel is turning anticlockwise:



In other words:

Anticlockwise Sequence	J10 (pin 12)	0	1	1	0	0	1	1 ...
	J11 (pin 11)	0	0	1	1	0	0	1 ...
Clockwise Sequence	J10 (pin 12)	0	0	1	1	0	0	1...
	J11 (pin 11)	0	1	1	0	0	1	1 ...

4-Player Adaptor (Team Tap)

The fact that 16 rows of data can be addressed allows a four-controller adaptor to be connected to **each** console controller port (for a total of 8 controllers using two adaptors). The 4-player adaptor is a device which expands either of the console controller ports to allow up to four controllers to be connected. It has four controller sockets (DB15 females, the same as on the console) for controllers to be connected, and a short cable with a DB15 male connector which plugs into the console.

The controller sockets on the adaptor have the 6 inputs wire OR’d together. The four output lines are an active low, 4 to 16 de-multiplexed version of the 4 console outputs.

Each socket recognizes four unique row codes which are used to specify requests for data from that controller. The table below shows the row codes which must be output from the Jaguar to request data from controllers connected to specific sockets on the adaptor. Note that socket 0 uses the same row codes as a single controller connected directly to one of the console controller ports.

Row Code Output From Jaguar:					Specifies which row of the controller is connected to:			
Port 2	J4	J5	J6	J7				
Port 1	J3	J2	J1	J0	Socket 0	Socket 1	Socket 2	Socket 3
	0	0	0	0		Row 0		
	0	0	0	1		Row 1		
	0	0	1	0		Row 2		
	0	0	1	1		Row 3		
	0	1	0	0			Row 0	
	0	1	0	1			Row 1	
	0	1	1	0			Row 2	
	0	1	1	1	Row 3			
	1	0	0	0			Row 3	
	1	0	0	1				Row 0
	1	0	1	0				Row 1
	1	0	1	1	Row 2			
	1	1	0	0				Row 2
	1	1	0	1	Row 1			
	1	1	1	0	Row 0			
	1	1	1	1				Row 3

Except for socket 0, the row codes shown in the table are not the row codes seen by the controllers

themselves. In order to make itself as transparent as possible to the controllers themselves, the adaptor converts the row codes for sockets 1-3 so that those controllers will only see socket 0 row codes.
In other words, when your program outputs the code %0101 that says it wants to read Row 1 of the controller connected to socket 2, the 4-player adaptor will convert the code to %1101 and then pass it to socket 2. The controller connected to socket 2 will see %1101, the same code you would use to access a single controller connected directly to the Jaguar, and return the appropriate information.

4-Player Adaptor and Advanced Controllers

Originally advanced controllers responded to row codes for socket 1 instead of the row codes for socket 0, this was to allow for a “pass through” connector into which a standard Joypad controller could be connected. They were then required to change their behaviour upon detection of a 4 Player adaptor, disabling the pass through and responding to socket 0 row codes themselves.

Due to advancements in Microcontroller design this has now changed (see the **Advanced Controllers** section for more information), consequently advanced controllers are no longer required to check for the presence of a 4-player adaptor (indicated by a +5V DC signal on pin 8 of each 4 Player adaptor socket), however they may still do so if the designer deems it necessary.

Because the 4 Player adaptor converts the row codes for sockets 1-3 to socket 0 row codes, only a controller read will be possible when Advance controllers are connected to a 4 Player adaptor. Software control of advanced features like rumble motors, force feedback and analogue/digital select will not be possible.

To summarize these ideas, the table below shows the various socket and controller positions with and without a 4-player adaptor (Ports 1 & 2 are identical in these respects).

Controller Port with 4-Player Adaptor			
Socket 0	Socket 1	Socket 2	Socket 3
The adaptor converts the row codes output by Jaguar programs and routes them to the appropriate socket. Socket 0 is the same as a controller plugged directly in the Port. Standard and Advanced controllers respond only to socket 0 row codes.			
Controller Port without 4-Player Adaptor			
A standard controller plugged directly in the port is the same as socket 0 of a 4-Player adaptor. Advanced controllers plugged directly into a port respond to socket 0 row codes for reads and socket 2 row codes for accessing configuration modes and advanced features.			

Bank Switching Controllers

Because there are 4 row codes allocated to each socket, the 4-player adaptor will only support 4 row controller devices. Without additional logic, each input supports up to 24 bits of data (4 rows of 6 bits). Three bits are reserved for the controller type identifier code, leaving 21 bits for data.

Intelligent controllers (i.e. ones that use a microcontroller), can multiplex even more data onto the same lines. One way this can be done is for the microcontroller to “Bank switch” whenever it sees a transition from row 3 back to row 0. Different bits of data are presented in each bank. See the section **Reading Bank Switching Controllers** later in this chapter for more information.

Detecting the 4-Player Adaptor & Connected Controllers

To detect the presence of a 4-player adaptor, a program should inquire the status of Row 1 of controller socket #3. If a 4-Player adaptor is present, the B0/B2 bit will be clear (0). Otherwise it will be set (1).

The pseudocode below demonstrates the basic technique for detecting a 4-player adaptor and the controllers connected to it, as well as any advanced controllers connected directly to the Jaguar.

```
For PORT = 1 to 2
      if PORT:SOCKET3:C1 = 0 then { 4-player adaptor found }
            for SOCKET = 0 to 3
                  PORT:SOCKET:CONTROLLERTYPE = PORT:SOCKET:C2/C3
                  if PORT:SOCKET:CONTROLLERTYPE = BANK-SWITHCING then
                        PORT:SOCKET:BANKSWITCHTYPE = DETECT_BANK_SWITCH_TYPE
                  end if
            next SOCKET
      else
            PORT:SOCKET0:CONTROLLERTYPE = STANDARD
            if PORT:SOCKET:C2/C3 = ROTARTY then
                  PORT:SOCKET1:CONTROLERTYPE = ROTARY
            else if PORT:SOCKET1:C2/C3 = BANK_SWITCHING then
                  PORT:SOCKET:BANKSWITCHTYPE = DETECT_BANK_SWITCH_TYPE
            endif
      endif
next PORT


FUNCTION_DETECT_BANK_SWITCH_TYPE
      DO
            READ ROWS 0, 1, 2, 3
      UNTIL ROW0:B0/B2 = 0 { Bank 0 }
      BANKCOUNT = 0
      DO
            READ ROWS 0, 1, 2, 3
            SAVE ROWDATA ( BANKCOUNT )
            BANKCOUNT = BANKCOUNT + 1
      UNTIL ROW0:B0/B2 = 0 { Bank 0 }
      return ROWDATA (BANKCOUNT - 1) : ROWS0-3:B1/B3
END FUNCTION
```

Caveats

The JOYSTICK and JOYBUTS registers return the same data in the same bits regardless of which socket is being read. However, be aware that without a 4-player adaptor, reading sockets 1-3 of a port may return an ‘echo’ of the standard Joypad controller at socket 0.
To avoid reading incorrect data, unless your program has detected that a 4-Player adaptor is connected, it should not try to read from sockets 1-3.

Advanced Controllers

General Information

All advanced controllers must contain a Microcontroller to act as their interface to the Jaguar console. Where the advanced controller uses analogue values, the Microcontroller should either utilise its own internal ADC to convert the analogue values to 8bit digital values or interface to a separate ADC chip that is also situated within the advanced controller.

Advanced controllers are required to conform to the following features...

1. When powered separately, controllers must use the +5 pin from the controller port to detect when the Jaguar is switched off and be held in Master Clear/Reset while the Jaguar is switched off.
2. Set their outputs to logic 1 at Jaguar power up to prevent issues with boot ROM controller reads.
3. Respond to socket 0 row codes for controller reads, socket 2 row codes for accessing configuration modes and advanced features.
4. Consume no more than 50mA of current from the Jaguar controller ports +5V pin and no more than 10mA for them to be suitable for games that support 3 or more players via the use of a 4-Player adaptor.
5. Have provision for connecting an external high current power source where the controller incorporates high current loads such as rumble motors or force feedback.
6. In the event that an advanced controller cannot house the required number of buttons internally (especially the critical Pause and Option buttons) it must have a DB15 female connecter fitted into which a standard Jaguar controller can be attached. The advanced controller is then required to read the standard controller and send that information to the Jaguar as one of its banks of output data (see **Reading Bank Switching Controllers**).
7. During the Jaguars “identifying connected controller” read advance controllers must output a special bank of data to allow a controller’s full identification and current configuration in a single bank read one pass. See the **Identifying Connected Controller Read** section for details of the bank output data.

To prevent any problems caused by row codes issued by the Boot ROM, advanced controllers should time the duration of the Socket 0, Row 0 codes and not output their initial row 0 data until they see a Socket 0, Row 0 code that is valid for at least 100µS. This is only necessary during the Jaguars “identifying connected controller” read, after that all row code timings are assumed to be correct for the type of controller being used (see the individual controllers’ specification/Analogue (Joystick and Driving) Controller section).

After the identifying connected controller phase it is recommended that advanced controllers issue controller data starting at Bank 0, however as you will see in the **Reading Bank Switching Controller** section that does not have to be the case as Bank 0 identification is included in the bank data, but doing so make it easier for the software programmer.

Reading Bank Switching Controllers

Most, if not all advanced controllers will output more than the single 24bits of data sent by the standard Joypad controller. Bank switching is a technique which allows a controller to return more information than would otherwise be possible with a single bank controller and is achieved by splitting the data into multiple Banks (groups) of 24bits, where one standard Joypad read is required per Bank of data. Consequently, unlike the standard Joypad controller the meaning of the 24bits sent to the Jaguar by bank switching controllers will change as each bank of data is read and is dependent on the controller type.

Bank switching is done automatically when the controller sees a transition from row 3 to row 0 (of the same controller socket). It is not possible to read only a particular bank or set of banks and ignore the other ones; you must always read all banks even if you don't really need all of the information.

Programs must always read an entire bank from a controller at once. However, it is not required that you read all banks from a single controller in a single pass. It is acceptable to read a bank from one controller, followed by a bank or multiple banks from other controllers, and then come back to read the next bank from the first controller.

The rows of each bank of a controller must be read in sequence: Row 0, Row 1, Row 2 and Row 3. The controller relies on the rows being read in sequence so that it can start processing the data for the next row in advance. The results of reading rows out of sequence are undefined; the data returned by the controller may be invalid. For example, you program would read data from an analogue joystick controller like this:

Bank 0:	**Row 0 *(controller will automatically bank switch here, then return Row0 data)* Row 1, Row 2, Row 3.**
Bank 1:	**Row 0 *(controller will automatically bank switch here, then return Row0 data)* Row 1, Row 2, Row 3.**

It is not necessary to know in advance which bank is active when you start reading. If you read all banks of a controller into a table, you can look at the data afterwards to figure out where the data for **Bank 0** is, and from there you can figure out where the data for the other banks must be. For example, if you were reading a driving controller, the data you read would end up in a table like this:

Bank 0							
Row 0		**Row 1**		**Row 2**		**Row 3**	
word 0	word 1	word 2	word 3	word 4	word 5	word 6	word 7
joystick	joybuts	joystick	joybuts	joystick	joybuts	joystick	joybuts
Bank 1							
Row 0		**Row 1**		**Row 2**		**Row 3**	
word 8	word 9	word 10	word 11	word 12	word 13	word 14	word 15
joystick	joybuts	joystick	joybuts	joystick	joybuts	joystick	joybuts

The bottom row of the table would be an array of WORD values read from the JOYSTICK and JOYBUTS registers. You could store these values into separate arrays if you prefer, and it is not necessary to read both the JOYSTICK register and JOYBUTS register for each row, but this example assumes you are always reading both registers and storing all the results into a single table for further processing.

In this example, Bank 0 came first, but that won't always be the case. You need to examine the data in the table to determine the location of each bank of data. Bank switching controllers always indicate **Bank 0** but setting bit 0 (**B0** of controller port 1) or bit 2 (**B2** of controller port 2) of the value read from the JOYBUTS register from row 0. The bit will be 0 for **Bank 0** and 1 for all other banks. Because the banks are always read in sequence, once you find **Bank 0** in the table, then you know where to find the data for all the other banks.

In the example above, because bit 0 of word 1 was clear (assuming controller port 1), then you would know that the data for **Bank 0** was in words 0-7. Since we only have two banks, that means the data for **Bank 1** must be in words 8-15.

Suppose you had a 6D controller, which has 3 different banks of information, connected to port 1. After reading 3 banks' worth of information from this controller, you might end up with a buffer that looks like this:

Bank 2							
Row 0		Row 1		Row 2		Row 3	
word 0	word 1	word 2	word 3	word 4	word 5	word 6	word 7
joystick	joybuts	joystick	joybuts	joystick	joybuts	joystick	joybuts
Bank 0							
Row 0		Row 1		Row 2		Row 3	
word 8	word 9	word 10	word 11	word 12	word 13	word 14	word 15
joystick	joybuts	joystick	joybuts	joystick	joybuts	joystick	joybuts
Bank 1							
Row 0		Row 1		Row 2		Row 3	
word 16	word 17	word 18	word 19	word 20	word 21	word 22	word 23
joystick	joybuts	joystick	joybuts	joystick	joybuts	joystick	joybuts

The first thing you need to do is find the data for **Bank 0**. First you would look at bit 0 of word 1, then word 9. In this example, word 9 would have bit 0 clear to indicate **Bank 0**. Therefore, words 8-15 contain the data for **Bank 0**. Once you know that, then you also know that **Bank 1** is contained in words 16-23 and **Bank 2** must be in words 0-7.

Note that there is a certain amount of processing time required when reading from one row to the next, because the microcontroller inside the controller has to put a different set of data on the outputs. This is normally approximately 10 microseconds (worse case is about 20 microseconds) when going from row to row within the same bank. Analogue controllers typically also require an additional 200 microseconds when going from one bank to the next (so that the analogue inputs may be digitalized). See the **Analogue Joystick and Driving Controllers** section for ideas on how to deal with this.

Identifying Connected Controller Read

The special output bank data for this identifying connected controller sweep is:

Controller ID Bank	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)*	1**	#Con1	#Con2	#Con3	#Con4
Row 2	0 (C2)*	1**	CurCon1	CurCon2	CurCon3	CurCon4
Row 1	1 (C1)*	1**	1	1	1	1
Row 0	1*	0**	1	1	1	1

* Bits B0/B2 shows values received for a Bank Switching controller where the C3 and C2 bits identify the basic controller type.

** When the C2 and C3 bits indicated a Bank Switching controller is connected, the B1/B3 bits identify the bank switching controller sub type (see table in the **Identifying Controller Types** section).

#Con bits indicate the number of different configurations the particular advanced controller supports, i.e. a mouse may have two configuration, two button and three button (see controller documentation for details).

CurCon bits indicate the default (current) controller configuration.

6D Controllers

These controllers support 6 degrees of freedom: Pitch, Yaw, Roll, X, Y and Z. We refer to Pitch as Z torque, Yaw as X torque and Roll as Y torque. Hence we have 6 values – X, Y, Z and TX, TY and TZ. We also define 7 buttons A-G.

Three banks of data are required, as we define 55 bits of information: 8 bit values for each of the 6 degrees of freedom (8*6 = 48 bits of information), plus 7 buttons.

Bank 0	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	D	X4	X5	X6	X7
Row 2	0 (C2)**	C	Z0	Z1	Z2	Z3
Row 1	1 (C1)	B	Y0	Y1	Y2	Y3
Row 0	0*	A	X0	X1	X2	X3

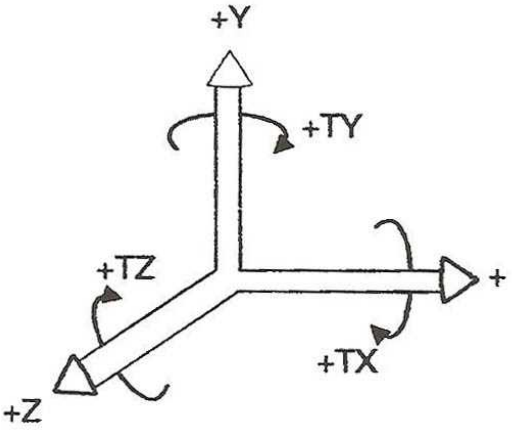
Bank 1	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	E	Y4	Y5	Y6	Y7
Row 2	0 (C2)**	F	TZ0	TZ1	TZ2	TZ3
Row 1	1 (C1)	G	TY0	TY1	TY2	TY3
Row 0	1*	Rezero	TX0	TX1	TX2	TX3

Bank 2	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	1**	Z4	Z5	Z6	Z7
Row 2	0 (C2)**	1**	TZ4	TZ5	TZ6	TZ7
Row 1	1 (C1)	1**	TY4	TY5	TY6	TY7
Row 0	1*	0**	TX4	TX5	TX6	TX7

* Bit B0/B2 of row 0 is used to synchronise the cycle of Banks. It will always be zero in Bank 0, while all other banks will return 1. Banks will cycle in the order Bank 0, Bank 1, Bank 2, Bank 0 etc. See **Reading Bank Switching Controllers** for more information.

** The C3 and C2 bits identify the basic controller type. The B1/B3 bits of the last bank of the controller are used to identify the specific bank switching controller type.

Value	Meaning
X(7:0)	X axis force
Y(7:0)	Y axis force
Z(7:0)	Z axis force
TX(7:0)	X axis, anticlockwise rotation torque
TY(7:0)	Y axis, anticlockwise rotation torque
TZ(7:0)	Z axis, anticlockwise rotation torque



X is positive right to left
Y is positive UP
Z is positive coming BACK (towards the user)

Torques are all positive in the COUNTER-CLOCKWISE direction, when facing the positive direction shown by the arrows above.

Reading a Joypad Controller Attached to An Advanced Controller

When an advanced controller reads an attached Joypad controller it should do so as if it were the Jaguar itself, in other words it should send the four Socket 0 row codes to the Joypad and read back the 24 bits of data as four rows of six data bits.

However, there is a slight difference in that the position of two of the data bits, Pause and C3 need to be switched before the Joypad data can be output as one of the advanced controllers' data banks as shown in the controller matrix below.

	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	Pause	Option	#	9	6	3
Row 2	1 (C2)	C	0	8	5	2
Row 1	1 (C1)	B	*	7	4	1
Row 0	1 (C3)	A	Up	Down	Left	Right

This is necessary to prevent any Bank 0 synchronisation issues (Pause presents as a false Bank 0 indicator) when reading the data banks into a table as described earlier.

Head Mounted Tracker

These devices provide three angular values, according to the orientation of the users head.

Bank 0	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	1	1	1	1	1
Row 2	0 (C2)**	1	AZ0	AZ1	AZ2	AZ3
Row 1	1 (C1)	1	AY0	AY1	AY2	AY3
Row 0	0*	1	AX0	AX1	AX2	AX3

Bank 1	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	0**	1	1	1	1
Row 2	0 (C2)**	1**	AZ4	AZ5	AZ6	AZ7
Row 1	1 (C1)	1**	AY4	AY5	AY6	AY7
Row 0	1*	1**	AX4	AX5	AX6	AX7

* Bit B0/B2 of row 0 is used to synchronise the cycle of Banks. It will always be zero in Bank 0, while all other banks will return 1. Banks will cycle in the order Bank 0, Bank 1, Bank 2, Bank 0 etc. See **Reading Bank Switching Controllers** for more information.

** The C3 and C2 bits identify the basic controller type. The B1/B3 bits of the last bank of the controller are used to identify the specific bank switching controller type.

Value	Meaning
AX(7:0)	Rotation angle around X (= roll = head tilted) axis
AY(7:0)	Rotation angle around Y (= yaw = looking left/right) axis
AZ(7:0)	Rotation angle around Z (= pitch = looking up/down) axis

Zero is facing straight ahead. Positive values are tilt left / look left / look up. Values are linear angle values, where +180 degrees = $7F, -179 degrees = $80.

Analogue (Joystick and Driving) Controller

These devices typically require 8 bits of analogue resolution in 2 dimensions (X and Y). Two 100K ohm linear potentiometers are typically used, with a +5V potential across the ends. The centre wiper will then read a voltage between 0 and +5.

To read this voltage requires an analogue to digital converter (ADC). A good solution is to use an ARM or Microchip microcontroller. Microcontrollers are available with varying numbers of ADC channels and general purpose digital I/O lines, i.e. a 16F73 or 18F24K20 have 6 ADC channels and 16 general purpose digital I/O lines. The four controller row outputs would be used to select one of four 6 bit addresses. The two 8 bit ADC values use 16 bits, leaving room for 5 switches and the 3 controller identifier codes.

In the example below, we have used bank switching to support even more switches. The bank is switched when the microcontroller sees a transition from Row 3 to Row 0. Bank identification is achieved by reading bits B0/B2 of Row 0. See **Reading Bank Switching Controllers** for more information.

Bank 0	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	D	Y4	Y5	Y6	Y7
Row 2	0 (C2)**	C	Y0	Y1	Y2	Y3
Row 1	1 (C1)	B	X4	X5	X6	X7
Row 0	0*	A	X0	X1	X2	X3

Bank 1	B2 B0	B3 B1	J12 J8	J13 J9	J14 J10	J15 J11
Row 3	1 (C3)**	1**	1	1	1	1
Row 2	0 (C2)**	1**	1	1	1	1
Row 1	1 (C1)	1**	1	1	1	1
Row 0	1*	1**	Up	Down	Left	Right

* Bit B0/B2 of row 0 is used to synchronise the cycle of Banks. It will always be zero in Bank 0, while all other banks will return 1. Banks will cycle in the order Bank 0, Bank 1, Bank 2, Bank 0 etc. See **Reading Bank Switching Controllers** for more information.

** The C2 and C3 bits identify the basic controller type. The B1/B3 bits of the last bank of the controller are used to identify the specific bank switching controller type.

	“Stick” Controller	“Driving” Controller
X (7:0)	Roll Right = Positive delta values from the centred position Left = Negative delta values from the centred position	Steering Right = Positive delta values from the centred position Left = Negative delta values from the centred position
Y (7:0)	Pitch Forward = Positive delta values from the centred position Backward = Negative delta values from the centred position	Accelerator/Break Accelerator = Positive delta values from the centred position Break = Negative delta values from the centred position
Up	Hat Switch “Up”	Gear shift Up
Down	Hat Switch “Down”	Gear shirt Down
Left	Hat Switch “Left”	Spare 1
Right	Hat Switch “Right”	Spare 2
A	Top Switch	Spare 3
B	Trigger Switch	Spare 4
C	Middle Switch	Spare 5
D	Lower Switch	Spare 6

The range of possible X and Y values is 0 -255, but not all controllers will use this entire range, and the range they do use is not predefined. Do not assume that certain constant values can always be used for the centre, hard right and hard left positions. Analogue devices are different from controller to controller, and even from day to day as temperature and humidity conditions change.

For example, a driving controller may return values of 160 (steering wheel centred), 245 (turned hard right) and 75 (turned hard left). A different controller of the same type from the same company (or the same controller under different temperature and/or humidity conditions) may return values of 150 (centred), 240 (hard right) and 55 (hard left). The centre position is different, and the value ranges are also different. Your software needs to be able to account for this.

It will be necessary to provide some sort of calibration routine where your program will ask the user to move the controller to certain positions, in order to read the values at those positions.[4] This should be an option on your controller configuration screen. It would also be nice if the user could choose to recalibrate the controller while paused in the middle of the game. It would be another nice touch if you stored the current calibration values into the cartridge EEPROM. That way, if the user is using the same controller under the same basic conditions most of the time, they won't be forced to recalibrate each time they play.

Analogue controllers require a certain amount of processing time from the time the row code is written to the JOYSTICK register until the data read back from the JOYSTICK or JOYBUTS registers will be valid. With a typical analogue controller, this delay is normally about 10 microseconds (worse case is about 20 microseconds) when going from row to row within the same bank (this delay applies to all Bank-switching controllers), and approximately 200 microseconds between banks.[5] There are two ways to handle this. You can do a small delay loop while waiting for the data to become available (do this in a way that uses the bus as little as possible, i.e. avoid memory access). Or if your program has a timer interrupt of some kind, you could write out the row code on one interrupt, and then wait for another interrupt before reading the value back. You could also use the GPU interrupts in a similar way. Whichever way you choose, try to avoid wasting CPU time and bus bandwidth just waiting to read the controller(s) when there is other processing you could be doing.

Advanced Controller Configuration Modes

When connected directly to a Jaguar controller port, advanced controllers will use Socket 2 row codes to enter and exit configuration modes as follow...
Socket 2, Row 0 = Enter Vibration/Force Feedback control mode
Socket 2, Row 1 = Exit Vibration/Force Feedback control mode
Socket 2, Row 2 = Enter configuration mode
Socket 2, Row 3 = Exit configuration mode

Row codes sent between the respective pairs of Enter and Exit row codes are used to set or turn on/off the respective features and should be detailed in the relevant controller documentation.

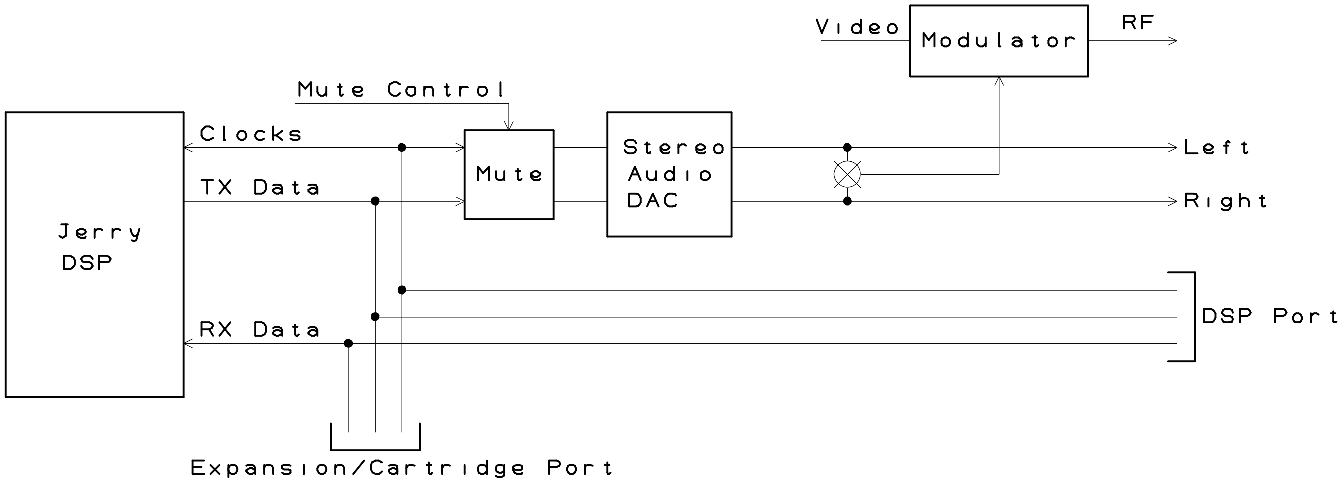
Keyboard/Mouse Interface

Note: The specifications for this controller type are still in the preliminary stages and are subject to change without notice.

[4] If you've ever played a game on a PC that used an analogue joystick, then you have probably seen examples of such calibration screens.

[5] These numbers were arrived at using a prototype Dual Shock controller interface comprised of a Microchip PIC Microcontroller running at 64MHz, consequently all Microcontrollers used in advanced controllers should run at a clock speed of 64MHz or higher for the given timings to be relevant.

Audio Subsystem



The Jaguar console includes a stereo 16 bit audio subsystem. Digital audio data can only be sourced from the Jerry DSP. This data can also be monitored at the expansion or DPS ports, on the TXD serial data line. Jerry can also read serial digital audio data on its RDX pin. The bit clock and word strobe signals can be sourced by Jerry, the expansion port or the DSP port. If the clock source is not Jerry then the software must force the Jerry clock lines tristate, by clearing bit 0 of SMODE.

The audio mute function has been added to allow non-audio serial data to be transmitted by Jerry without making a horrible noise on the audio outputs. When serial peripherals are connected to the DSP port, and are in use, the audio should be muted by writing zero to bit 8 of the JOYSTICK register ($F14000).

Cartridges and NVRAM

The Jaguar console cartridge port supports up to 6 Megabytes of space. Cartridges can be 8, 16 or 32 bit wide.[6] Special support is also included for serial EEPROMS. Reading and writing the EEPORM **must** be done through the Atari supplied routines. (See the sample program for accessing NVRAM.) This is the only way to ensure reliable operation.

Bit 0 of the JOYSTICK register, when read, represents the data output bit of the EEPROM, and **not** the J0 input from the joystick. Since J0 has always been used as an output so far, this should not cause problems. But bear in mind that this data bit is now random when read, and not equal to the J0 output bit as before.

It should be noted that the EEPROM uses addresses in the GPIO0 and GPIO1 range ($F14800-$F15FFF). Any inadvertent access (reads or writes) to these address ranges will cause subsequent EEPROM reads and write to fail. So don't do it...

When you build your own 32 bit test cartridges using Atari's 4-chip EPROM cartridge blanks, the ordering of data in the chips is as follows:

Chip	Bytes	Bits in 32-bit long
U1	$800003, $800007, $80000B, etc.	d0-d7
U2	$800002, $800006, $80000A, etc.	d8-d15
U3	$800001, $800005, $800009, etc.	d16-d23
U4	$800000, $800004, $800008, etc.	d24-d31

In a non encrypted test cartridge, locations $800000 to $801FFF should have values of $FF. Your program code should always start at $802000 in both encrypted and non encrypted cartridges.

Burning Your Own Cartridge EPROMs

For those wanting to use an EPROM burner to create their own non-encrypted test cartridges, any EPROM burner capable of handling 4 megabit EPROM chips should be acceptable.

If you would like a recommendation for a particular EPROM burner, Atari has had good success with the **Pilot** EPROM burners, manufactured by **Advin**.

6 At this time, the Stubulator ROM used in development machines currently only supports the use of 32-bit wide cartridges.

EPROMs For Making Cartridges (game code)

The following EPROM types have been successfully used in Atari's test department:

1 Megabyte cartridge:
For a 4x4 EPROM cartridge with128 bytes EEPROM, a cartridge uses (4) 512Kbit x 8 (4 megabit) chips:

Manufacture	Chip code
Macronix	MX27C4000DC-12
Toshiba	TC574000AD-120 or TC574000AD-150
AMD	AM27C040-150DC
	27C400

Chips with access speeds slower than those shown above are not recommended. Similar chips from other manufactures may work, however have not been tested by Atari. Try them at your own risk.

2 Megabyte cartridge:

For a 16x2 EPROM cartridge with128 byte EEPROM, a cartridge uses (2) 512Kbit x 16 (8 megabit) chips:

Manufacture	Chip code
	27C800

4 Megabyte cartridge:

For a 16x2 EPROM cartridge with128 byte EEPROM, a cartridge uses (2) 1024Kbit x 16 (16 megabit) chips:

Manufacture	Chip code
	27C160

6 Megabyte cartridge: (modified Atari PCB's and new non Atari PCB's)

For a 16x2 EPROM cartridge with128 byte EEPROM, a cartridge uses (2) 2048Kbit x 16 (32 megabit) chips:

Manufacture	Chip code
	27C322

EEPROMs For Making Cartridges (game saves)

Manufacture	Chip code	Size	Comments
	93c46b	128 byte	Suitable for Atari PCBs & duplicates
	93c46c	128 byte	Suitable for alternate PCB designs
	93lc46c	128 byte	Suitable for alternate PCB designs
	93c86b	2K byte	Suitable for Atari PCBs & duplicates
	93c86c	2K byte	Suitable for alternate PCB designs
	93lc86c	2K byte	Suitable for alternate PCB designs

Appendixes

Jaguar Power Supply Specifications

From new both the Jaguar console and the CD unit are supplied with a power supply unit (PSU) that plugs straight into your wall outlet. This PSU is rated to supply an unregulated 9 Volts DC at 1.2 Amp (1200mA) via a 2.1mm DC power connector wired Outer positive (+, 9V) and Inner or Centre/Tip negative (-, 0V).

Should you need to replace such a unit and cannot find an original Jaguar replacement there are two types of third party replacement available, regulated (a.k.a. stabilized) and unregulated. To prevent damage to your Jaguar it is imperative that any unregulated replacement matches the voltage, current and polarity specifications mentioned above exactly. Regulated types must match for voltage and connection type/polarity and may supply more (but not less) then 1.2 Amps of current.[7]

Standard Jaguar Current Consumption (Game + 2 JoyPads) = 560mA (Average, but can go to 800mA)
Standard Jaguar CD unit Current Consumption = 650mA (Peak)

This document recommends various limits for the amount of current that can be drawn from the Jaguar's controller, expansion (cartridge), video and DSP ports, **these should not be exceeded**. With the standard Jaguar power brick only capable of supplying 1.2A (1200mA) of current it is not possible to power both the Jaguar and the CD unit from a single standard power brick.

Reading a Jaguar Controller Supplemental (Atari Source code issue)

You may have noticed in the main **Reading a Jaguar Controller** section that the lower 8 bits sent to the JOYSTICK register are a palindrome, i.e. %01111110 or %10011001. This is to allow you to read data from the same row of the same socket on both controller ports at once. Anyone using any of the Atari source code for controller reads may have noticed that the code provided is written to read and assemble the data from one controller port only, thus requiring you to issue the row codes for a particular socket twice to read both controller ports.

While there is nothing wrong with this in theory, in practice it would be disastrous where advanced controllers are concerned as the second issue of the row 0 codes would cause them to bank switch. As a result every time you read a controller port (assuming you read both) you would be reading every other bank of an attached advanced controller thus causing all kinds of bank synchronisation and missing data problems.
There are two solutions to this...

1) Write you own controller read code that reads and assembles the controller data from both controller ports at once (requires less controller row reads) or
2) Change either the low or high nibble of data sent to the JOYSTICK as applicable so that is it no longer a palindrome, using dummy data for the 4 row bits of the controller port not being read. For example, instead of issuing the palindromic socket 0 row 0 code %01111110 twice you issue $11111110 when reading controller port 1 and $01111111 when reading controller port 2.

7 If there is no clear indication that your chosen replacement PSU is regulated (stabilised) then assume it is unregulated. This can be checked by measuring its output voltage with a multi-meter or DVM while disconnected from the Jaguar, if its output voltage is 9V then it is regulated, if it is higher, typically 12V - 14V then it is unregulated.

Jaguar Cartridge Parts (provided for information only, other sources may be available)

PCBs:
New unpopulated 4 & 6Mb Jaguar PCBs can be purchased from Clint Thompson, to order either PM him on Atari Age or e-mail info@riscgames.com.

In addition to the EPROMS you will also need to purchase 0.1uF (100nF), 16V multilayer ceramic/ceramic plate 0.2"(200thou/5mm) pitch decoupling capacitors separately.

Shells (Cases):
You can purchase new cases in several colours from Atari Age member Starwander' website.

Information was correct at the time of writing.

Jaguar Schematic Diagram

Click here to open a clear version of the Jaguar schematic PDF file.
Click here to open a clear version of the Jaguar schematic png file.

CD Unit schematic diagram

Click here to open the schematic PDF file.

Standard Jaguar Controller Supplemental (Schematic Diagram)

Click here to open the schematic PDF file.

Notes:
Indicated wire colours may not be the same in all controllers.
Controller ID diodes are not required as C2 and C3 in the controller matrix are high to indicate a standard controller.
However, diode D21 is included for refence only, to indicate how the Bank Switching controller is identified (pulls C2 low).

Pro Jaguar Controller schematic diagram

Click here to open the schematic PDF file.

Notes:
Indicated wire colours may not be the same in all controllers.
The additional buttons are simply connected in parallel with the standard equivalents.

Team Tap Supplemental (Schematic Diagram)

Click here to open the schematic PDF file.

Notes:
Indicated wire colours may not be the same in all controllers.
Diode D25 identifies the Team Tap to the Jaguar by pulling J1 of socket 3 (D_J1 in the schematic, C1 in the controller matrix) low (0). For all other Team Tap sockets and all other controller types C1 will be high (1).

Rotary “Tempest” Controller Supplemental (Schematic and Identification)

Click here to open the schematic PDF file.

Notes:
Indicated wire colours may not be the same in all controllers.
Diode D21 (C3 in the controller matrix) is show for refence to indicate how the rotary controller is identified.

All currently available Rotary controllers are modified standard controller and most still identify themselves as a standard controller. This would be a problem with software that attempts to correctly identify them. The Tempest 2000 compatible solution is a relatively simple one requiring the addition of one 1N4148 silicon diode and can be done by almost anyone as follows...

1) Turn the controller over and remove the four rubber pads.

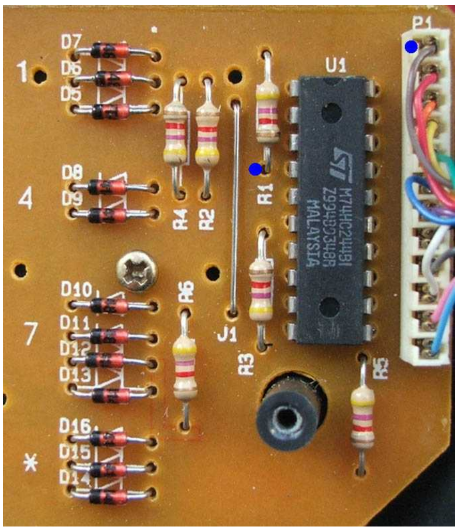
2) Using a medium size Phillips (Crosshead) screwdriver remove the four screws holding the case together and remove the underside of the case.

3) With the underside of the case removed you will see two circuit boards, holding the controller with the cable end away from you the diode should be fitted between the lower end of R1 and the top pin of the connector on the right side of the PCB closest to you as indicated by the blue dots in the image below (left).

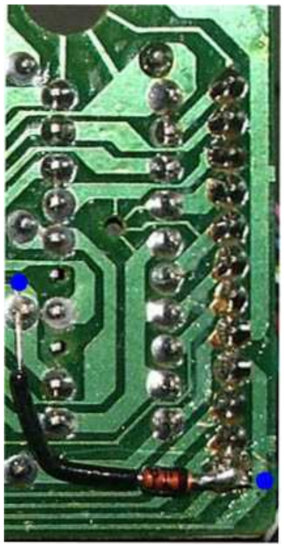
4) The easiest way of doing this is to attach the diode on the solder side of the PCB. To do this you first remove the two screws holding in the PCB and carefully turn the PCB over flipping it away from you 180 degrees. R1, the IC and the connector will now be in the bottom right of the PCB. Solder the diode on the solder side of the PCB to the points indicated by the blue dots in the image below (right). Make sure to connect the cathode of the diode (end with the black line) to the connector and not to R1.

5) Although there is very little for the diode to short against, I have insulted the longer leg that is connected to R1, I used a section of PVC insulation from a wire but any non-conductive tape such as masking, electrical or even Sellotape™ stuck to the PCB under the diode leg will suffice.

6) Reassemble your controller.



Components Side view


Solder Side View