Odyssey 2 Technical Specs V1.1

By Daniel Boris 11/98

Disclaimer: All the information in the original document come from studying the actual Odyssey 2 hardware. The only technical documentI had that the time was the Intel 8-bit Embedded Controller Handbook. Starting with version 1.1, I had access to various official O2 tech docs. I have made every attempt to assure the accuracy of this information, but there are bound to be errors and omissions in this document. ***Use this information at your own risk.***

1.0 Processor

The Odyssey 2 (O2) is based on the Intel 8048 microcontroller. The 8048 is clocked at approximately 1.79 MHz which is divided by 5 to produce a .36Mhz (2.5us) instruction cycle clock. (Note: This is not as slow as it seems since each 8048 instruction takes only 1 or 2 cycles) The 8048 has 64 bytes of internal RAM, and 1K of internal ROM that contains the system BIOS (see section 6.0). The 8048 has 2, 8-bit I/O ports, an internal timer/counter, an interrupt input, and 2 single bit testable inputs.

1.1 I/O port 1

All the pins (P10-P17) on this port are used as outputs. The function of each pin as follows:

P10: Cart bank switch 0

This pin is used to select between rom banks on 2 bank cartridges and is the low bit for selecting banks in 4 bank cartridges.

P11: Cart bank switch 1

This pin is the high bit used for selecting banks in a 4 bank cartridge.

P12: Keyboard scan enable

Setting this pin to 0 enables the output of the keyboard scanner.

P13: Video Display Controller (VDC) enable

Setting this pin to 0 enables the video controller onto the bus.

P14: External RAM enable

Setting this pin to 0 enables the external ram onto the bus.

P15: Not connected

P16: Copy mode enable.

Setting this bit to 1 enables the RAM to VDC copy mode. To use this mode you must also set P13 and P14 to 0. In this mode all external reads come from the external ram and all external writes go to the VDC. This allows data to be copied from the RAM to the VDC easily.

P17: Luminance enable.
I believe setting this bit to 1 enables the luminance output from the VDC to the video mixer.

1.2 I/O port 2

The second I/O port is used to read the keyboard. Since P20-P23 are also used as the upper 4 bits of the address bus, I believe that it is only possible to read the keyboard correctly through the BIOS.

P20..P22(W): Select keyboard row to scan
The value written into these 3 bits will pull one of the keyboard rows low (see Appendix A). Writing a value of 0 will also enable joystick 2 onto the data bus, and writing a value of 1 will enable joystick 1 onto the data bus (see section 5.0).

P23: Unused.

P24(R): Key press indicator
A 0 on this input indicates that a key was pressed on the currently selected row.

P25..P27(R): Keyboard column read.
If R24=0 then these 3 bits will contain the column number of the key that was pressed.

2.0 Cartridge

The O2 cartridge connector has 2 row of 15 contacts each. The contacts are on 0.15 inch centers. The cartridge port also serves as an expansion port, for example for the Voice Module. The pins are labeled as follows:

Top Row:	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R
Bottom Row:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

(Note that letters I, O, and Q are not used as labels on the top row)

A: ~WR – Active low cart write enable, disabled when P16 is high.
B,C: GND
D: +5V
E: Active high external RAM enable
F: ~PSEN (Chip Enable to the cartridge)
G: A0
H: A1

J: A2
K: A3
L: A4
M: A5
P: A6
N: A7
R: A8

1: T0 from processor
2..9: DB0..DB7 from the processor
10: A10
11: P14
12: P11, High bank select bit
13: P10, Low bank select bit
14: A11
15: A9

A normal cartridge uses:
GND, +5: Power

DB0..DB7: Data Bus

~PSEN (Program Store Enable): This signal comes from the processor and goes low to indicate a read from program memory. This is used to enable the cartridge.

A0...A9,A11: Address bus. Notice that A10 is not used by the cartridge. The reason for this is that the internal ROM in the processor occupies address $0-$3FF. The first byte in the cartridge is accessed at address $400 so leaving off A10 will map this address to the first byte in the cartridge.

P10, P11: Bank select bits. For cartridges over 2K these lines are used to select banks. For 4K carts P10 is used to select between the 2 banks. For 8K carts P10 and P11 are used to select between the 4 banks.

3.0 External RAM

Besides that RAM that is internal to the processor there are 128 bytes of RAM external to the processor. To enable the external ram set P14 low and P13 high. The memory can now be read and written to using the MOVX command.

4.0 VDC

The Video Display Controller (VDC) chip is a custom Intel IC that generates all the O2’s video and audio. The VDC is accessed by setting P13 low and P14 high, then using the MOVX instruction. The VDC occupies and address range from $00 - $FF. It is important to note that the VDC registers that control the graphic object cannot be changed while the VDC is enabled by the VDC control register.

4.1 Graphic Objects

The VDC can generate 4 different types of graphics ‘objects’, a background grid, single characters, quad characters, and sprites. Under normal circumstances, these objects should only be changed during the vertical blank period.

4.2 Background Grid

The background grid is a grid of boxes with 8 rows and 9 columns. Each segment of each of the grid lines can be individually turned on or off. The VDC registers that control the grid are at locations $C0-$C8, $D0-$D8, $E0-$E9:

$C0-$C8 Horizontal grid line (0=off/1=on)

Each location represents 1 of the first 8 horizontal lines in the grid. Each bit in each location controls the corresponding column on each line, Bit 0 = column 0, Bit 1= column 1 and so on.

$D0 - $D8 Horizontal grid line 9 (0=off/1=on)

These locations work the same as $C0-$C9 but for horizontal line 9. Only bit 0 is used in each location.

$E0-$E9 Vertical gird lines (0=off/1=on)

Each location represents 1 of the 10 vertical grid lines. Each bit in each location controls the corresponding row on each line, Bit 0 = row 0, Bit 1 = row 1 and so on.

The color of the background and grid are controlled by the color register $A3 (see section 4.9).

If bit 7 of the VDC control register ($A0) is set to 1 then the box to the right of each vertical line segment that is turned on will be filled solid with the grid color.

4.3 Sprites

The VDC can draw 4 independent sprites. Each sprite is 8x8 pixels in one color and can be positioned freely anywhere on the screen. The sprite control registers are at $00 - $0F and the sprite shape memory is at $80-$9F.

$00 - $03 Sprite 0 control
$04 - $07 Sprite 1 control
$08 - $0B Sprite 2 control
$0C - $0F Sprite 3 control

$80 - $87 Sprite 0 shape
$88 - $8F Sprite 1 shape
$90 - $97 Sprite 2 shape
$98 - $9F Sprite 3 shape

4.3.1 Sprite control registers

Byte 0 – Y position of sprite

Byte 1 – X position of sprite

Byte 2:

- Bit 0: 1 = Shift sprite 1 pixel to the right
- Bit 1: 1 = Shift even rows of sprite 1 pixel to the right
- Bit 2: 0 = Normal sprite/1=Double size sprite
- Bit 3..5: Sprite color
- Bit 6..7: Unused

Byte 3: Unused

4.3.2 Sprite data

Each sprite has 8 bytes for the shape. Each byte represents one row of the sprite and each bit controls one column of the sprite.

4.4 Single Characters

The VDC can generate up to 12 foreground characters from it's internal character set of 64 characters (see Appendix C). The memory area $10-$3F controls these characters and each character requires 4 bytes so the first character is at $10-$13, character 2 is at $14-$17, etc. Note that if any characters over lap each other, or overlap quad characters, the resulting display will be unpredictable.

Byte 0: Y position to start displaying character at.

Byte 1: X position of character

Byte 2: Lower 8 bits of position in character set to begin getting character shape from.

Byte 3: Bit 0: 9th bit of character set pointer.
Bit 1..3: Character color

NOTE: The interaction between Byte 0 and Byte 2 in determining how and where the character is displayed is very confusing and hard to explain in words. The best way to learn it is to write some code and play around with it.

4.5 Quad characters

Quad characters work in a very similar fashion to normal characters except that they are displayed in groups of 4 characters at a time. The quad characters use the memory range from $40-$7F, and there are 16 bytes for each quad character. In each quad character object there are 4 bytes for each of the 4 characters and they have the same functions as the 4 bytes in a normal character. The only difference is that the X position and Y position of the last character sets the position of the whole set of 4 characters. Each character is displayed one after the other with a one character wide space between each. Just like normal characters, if any quad characters overlap each other, or overlaps normal characters the results will be unpredictable.

4.6 VDC Control Register

Location $A0 is the VDC control register. The bits in this register have the following functions:

Bit 0: Enable horizontal interrupt. Enables an interrupt at each horizontal blank.
Bit 1: 1 = Strobe X,Y beam location into $A4,$A5. 0 = $A4,$A5 follow beam.
Bit 2: Enable sound interrupt.
Bit 3: Grid control (0=Off/1=On)
This bit turns the background grid on or off. Grid cannot be changes when this is set to 1.
Bit 4: Unused in the O2.
Bit 5: Foreground (0=Off/1=On)
This bit turns all foreground objects on or off. Characters and sprites cannot be changed when this is set to 1.
Bit 6: Dot Grid (0=Off/1=On)
This bit turns a gird of dots on or off. These dots line up with the intersections of the grid lines in the background grid.
Bit 7: Fill Mode (0=Off/1=On)
This bit turns the grid fill mode on or off. When this bit is on, the box to the right of every vertical grid line that is turned on is filled with the background color.

4.7 VDC Status register

Location $A1 is the VDC status register:

Bit 0: 1 = Horizontal scan active, 0 = HBLANK
Bit 1: Position Strobe Status. 1 = Follow Beam, 0 = Latched
Bit 2: Sound register empty
Bit 3: This bit is normally 0 and goes to 1 for 40us at the start of VBLANK
Bit 4..5: Unused
Bit 6: External Chip Overlay (not used on the O2)
Bit 7: Character overlap. This is set when 2 or more character objects are overlapping

4.8 Collision register

Location $A2 is the collision register. Each type of object on the screen has a corresponding bit in the collision register:

Bit 0: Sprite 0
Bit 1: Sprite 1
Bit 2: Sprite 2
Bit 3: Sprite 3
Bit 4: Vertical grid
Bit 5: Horizontal grid and dots grid
Bit 6: External collision input, not used on O2.

Bit 7: Characters

To check for collisions with a given object(s) set the appropriate bits to 1. When you read back this location any collision between objects will be indicated by a 1 in the corresponding location. For example to check for collisions between sprite 0 and other objects, set $A0 to $01, then when you read back $A0 it will return a 1 for each object sprite 0 collided with. This register is only valid during VBLANK.

4.9 Color register

Register $A3 is the register that controls the background and grid color. The bits have the following functions:

Bit 0..2: Grid color
Bit 3..5: Background color
Bit 6: Grid luminance (0=dim/1=bright)
Bit 7: Unused

4.10 Sound

The sound is controlled by registers $A7 to $AA. The sound system consists of a 24 bit shift register that is clocked out at one of two frequencies to form the output audio signal.

$A7-$A9 – 24 bit shift register
$AA – Sound control register

Bit 0-3: Sound volume
Bit 4 : Enable noise generation.
Bit 5 : Shift frequency, 0 = 983Hz, 1 = 3933Hz.
Bit 6 : Recirculation bit. 0 = Shift register is played once and sound stops, 1 = Shift register is constantly looped.
Bit 7 : Sound enable. 0 = sound off, 1 = sound on

4.11 Video timing (approximate timings)

Video frame = 16.6ms = 262 scanlines = 6026 machine cycles
VBLANK = 1.4ms = 22 scanlines = 506 machine cycles
Display area = 15.2ms = 240 scanlines
Scanline = 63.4us (including HBLANK) = 23 machine cycles
HBLANK = 12us

4.12 *(Removed, see section 4.14)*

4.13 Line Interrupts

In normal operation video registers should only be changed during the vertical blank period. Video registers can be changed during the screen drawing with some restrictions to create desirable effects. To synchronize these changes you can use line interrupts.

The 8048 processor has an internal counter that is clocked from the horizontal sync signal from the VDC. At the end of each horizontal blank period the internal counter is incremented, when the counter rolls over from $FF to $00 a counter interrupt is generated.

Some cartridges do mid screen changes without using line interrupts. To do this they use the scanline counter ($A4) to determine when to make changes.

4.14 X,Y Regisers

$A5 and $A4 are the X and Y registers respectively. These registers contain the current X and Y position of the electron beam as it drawing the screen. When register $A0 bit 1 is set to 1 these positions will be latched in and held in these registers.

5.0 Joysticks

To read the joysticks you must first disable both the VDC and the external RAM by setting P13 and P14 high. You then enable the keyboard scanner by setting P12 low. To read the right joystick set P20...P22 to 0 and to read the left joystick set P20, P21 to 0 and P22 to 1. Once this is setup the stick can be read using the INS A,BUS command. The bits read are as follows:

Bit 0: Up
Bit 1: Right
Bit 2: Down
Bit 3: Left
Bit 4: Fire button

NOTE: Since the joystick is enabled onto the data bus it is only possible to read the sticks from the BIOS ROM.

6.0 BIOS

The O2 has a 1K BIOS ROM that is stored inside the 8048 processor. The BIOS takes locations $0-$3FF in the program area.

6.1 Cartridge Vectors:

The BIOS ROM makes calls to various addresses in the cartridge:

$400 – The BIOS jumps to this address when the systems is powered up or reset.
$402 – Jumped to on an external IRQ.
$404 – Jumped to on a timer IRQ.
$406 – Vblank service routine
$408 – The BIOS jumps to this location after the “Select Game” routine has executed.
$40A – Continuation of Vblank

6.2 BIOS Routines: (This section is far from complete)

Alters: Indicates any registers that are altered by the routine and not restored
Ends: Ending address of routine
Inputs: Indicates any values that should be passed to the routine
Returns: Indicates any values that the routine returns

$E7: Enable VDC
  Alters: P1
  Ends: $EB

$EC: Enable external RAM
  Alters: P1
  Ends: $F0

$11C: Turn display off
  Alters: R0,A

$127: Turn display on
  Alters: R0,A

$2C3: Select Game routine. (This is where most cartridges start)

$38F: Read Joystick
  Input: R1: 0 = Read stick 0, 1 = Read stick 1
  Returns: R3 = $FF Stick pushed up
      R3 = $01   Stick pushed down
      R2 = $FF Stick pushed left
      R2 = $01 Stick pushed right
      F0 = 1   Button pushed
  Alters: A, P2, R0, R2, R3, F0

7.0 The Voice

The Voice is an expansion module that plugs into the O2's cartridge port and give it voice synthesis capabilities. The voice uses an SP0256B speech processor chip with built in speech ROMs. The speech ROM can be expanded via the cartridge connector so that carts can contain extended voice data.

7.1 Addressing The Voice

The Voice appears in memory just about the external RAM. It is accessed the same way that the external RAM is accessed (see section 3.0). Any external writes to address $80-$FF will go to The Voice. Setting bit 5 to 0 when writing to the address range will hold the SP0256 is reset. Setting bit 5 to 1 will release the reset. Reset will stop any sound that is currently playing and will reset the voice bank select back to bank 0.

7.2 Voice Banks

The various sounds that the SP0256 can play are store in a series of different banks. The first bank is contained inside the SP0256 chip and is selected by writing to address $E4. There are 3 additional banks of sound on an external ROM board within The Voice. These banks are selected byte writing to $E8, $E9, and $EA. Writing to addresses $EB-$EF will select external sound banks that are contained on some cartridges. Once a bank is selected it stay selected until another bank is selected, or until the SP0256 is rest.

7.3 Triggering sounds

Writing a 1 to any address from $80-$DF, $F0-$FF will trigger the SP0256 to play a sound from the current bank. Note, not all banks will contain sounds at all locations. See Appendix E for a list of sounds. Once a sound is triggered it will play until it is completed, or until the SP0256 is reset.

7.4 Voice Status

The T0 input to the processor comes from the LRQ pin on the SP0256. Whenever LRQ is 1 the input buffer to the SP0256 is full and no more sounds commands can be issued. When LRQ goes low the input buffer is free to be loaded. Note that this does not indicate that a sound has finished playing, it just indicates that the chip is read to receive another command. New commands will not execute until previous ones have finished executing.

Appendix A: Keyboard map

					READ			
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	2	3	4	5	6	7
1	8	9			SPC	?	L	P
2	+	W	E	R	T	U	I	O
3	Q	S	D	F	G	H	J	K
4	A	Z	X	C	V	B	M	.
5	-	*	/	=	Y(YES)	N(NO)	CLR	ENT

The reset button is hardwired to reset on the processor and cannot be read like the other keys.

Appendix B: Colors

	Character/Sprite Colors	**Dark Back/Grid Colors**	**Light Back/Grid Colors**
0	Dark Grey	Black	Black
1	Red	Dark Blue	Blue
2	Green	Dark Green	Green
3	Orange	Light Green	Light Green
4	Blue	Red	Red
5	Violet	Violet	Violet
6	Light Grey	Orange	Orange
7	White	Light Grey	Light Grey

Appendix C: Character Set

	Character		**Character**		**Character**		**Character**
0	0	16	+	32	A	48	10
1	1	17	W	33	Z	49	“ball”
2	2	18	E	34	X	50	“man right”
3	3	19	R	35	C	51	“man right walk”
4	4	20	T	36	V	52	“man left walk”
5	5	21	U	37	B	53	“man left”
6	6	22	I	38	M	54	“arrow right”
7	7	23	O	39	.	55	“tree”
8	8	24	Q	40	-	56	“slope left”
9	9	25	S	41	x	57	“slope right”
10	:	26	D	42	“divide sign”	58	“man forward”
11	$	27	F	43	=	59	\
12		28	G	44	Y	60	“ship 1”
13	?	29	H	45	N	61	“plane”
14	L	30	J	46	/	62	“ship 2”
15	P	31	K	47	“block”	63	“ship 3”

Appendix D: VDC memory map

$00 - $03	Sprite 0
$04 - $07	Sprite 1
$08 - $0B	Sprite 2
$0C - $0F	Sprite 3

Byte 0
Y position

Byte 1
X position

Byte 2							
7	6	5	4	3	2	1	0
		Color			Size	Even Shift	Full Shift

$10 - $3F	Foreground Characters (12 characters/4 bytes per character)

Byte 0
Y position to start displaying character

Byte 1
X position

Byte 2
Character set pointer (0..7)

Byte 3							
7	6	5	4	3	2	1	0
				Color			Cset pointer (8)

$40 - $7F	Quad Characters (4 groups/16 bytes per group)

$80 - $87	Sprite 0 shape
$88 - $8F	Sprite 1 shape
$90 - $97	Sprite 2 shape
$98 - $9F	Sprite 3 shape

$A0	VDC control register

7	6	5	4	3	2	1	0
Fill Mode	Dot Grid	Display Enable	Enable Ext. Overlap	Grid on/off	Sound Interrupt Enable	Latch Pos Registers	Horz Interrupt Enble

$A1	VDC Status

7	6	5	4	3	2	1	0
Major System Overlap	External Overlap			VBLANK	Sound Needs Service	Position Strobe Status	HBLANK

$A2	Collision (W = enable/R = detect)

7	6	5	4	3	2	1	0
Characters			Grid	Sprite 3	Sprite 2	Sprite 1	Sprite 0

$A3	Color register

7	6	5	4	3	2	1	0
	Grid Lum	Background Color			Grid Color		

Appendix E: Voice sounds

```
Bank: $E4

˝
80  10MS PAUSE
˝
81  30MS PAUSE
˝
82  50MS PAUSE
˝
83  100MS PAUSE
˝
84  200MS PAUSE
˝
85  b[OY]
˝
86  sk[Y]
˝
87  [E]nd
˝
88  [C]OMB
˝
89  [P]ow
˝

8A  dod[GE]
˝
8B  thi[N]
˝
8C  S[i]T
8D  [T]o
8E  [R]ural
8F  s[U]cceed
90  [M]ilk
91  par[T]
92  [TH]ey
93  s[EE]
94  b[EI]ge
95  coul[D]
96  t[OO]
97  [AU]ght
98  h[O]t
99  [Y]es
9A  h[A]t
9B  [H]e
9C  [B]usiness (Short)
9D  [TH]in
9E  b[OO]k
```

9F	f[OO]d
A0	[OU]t
A1	[D]o
A2	wi[G]
A3	[V]est
A8	[F]ood
A9	s[K]y
AA	[C]an't
AB	[Z]oo
AC	a[NG]chor (anchor)
AD	[L]ake
AE	[W]ool
AF	[R]epair
B0	[WH]ig
B1	[Y]es
B2	[CH]urch
B3	f[IR] (Short)
B4	f[IR] (Long)
A4	[G]ot
A5	[SH]ip
A6	a[Z]ure
A7	b[R]ain
B5	b[EAU]
B6	[TH]ey
B7	ve[S]t
B8	[N]o
B9	[H]oe
BA	st[ORE]
BB	al[AR]m
BC	cl[EAR]
BD	[G]uest
BE	sadd[EL] (saddle)
BF	[B]usiness (Long)
C0	"ENEMY"
C1	"ALL CLEAR"
C2	"PLEASE"
C3	"GET OFF"
C4	"OPEN FIRE"
C5	"WATCH OUT"
C6	"MERCY"
C7	"HIT IT"
C8	"YOU BLEW IT"
C9	"DO IT AGAIN"
CA	"INCREDIBLE"
FA	"U.F.O."
FB	"MONSTER!"

Bank: $E8

80	"AMAZING"
81	"THANK YOU"
82	"YUCK"
83	"ARG"
84	"THAT'S EASY"
85	"ATTENTION"
86	"DANGER"
87	"TURKEY"
88	"ACTION"
89	"AAAAH"
8A	"GOOD"
8B	"HI"
8C	"HARD"
8D	"RIGHT"
8E	"WRONG"
8F	"ATTACK!"
90	"GREAT!"
91	"CLIMB"
92	"DIVE"
93	"FIRE"
94	"HELP"
95	"HURRY"

Bank: $E9

80	"JUMP"
81	"RUN"

82	"SQUASH!"
83	"NOW"
84	"NO"
85	"NO!"
86	"YES"
87	"SORRY"
88	"OH DEAR"
89	"GOT'CHA"
8A	"OUTCH"
8B	"OH NO"
8C	"GO FOR IT"
8D	"DO IT"
8E	"LOOK OUT"
8F	"COME ON!"
90	WHIP SOUND
91	TONE E991
92	MEDIUM EXPLOSION
93	BEE SOUND
94	SMALL EXPLOSION E994
95	SMALL EXPLOSION E995
96	SMALL EXPLOSION E996

Bank $EA

80	"UP"
81	"DOWN"
82	"ALAS"
83	"GO"
84	"FIGHT"
85	"DODGE"
86	PYEW-THUMP
87	PYEW-PYEW
88	MISSILE
89	LARGE EXPLOSION
8A	"HA HA HA HA HA"
8B	BRRRREEP
8C	GUNSHOT
8D	TONE EA8D
8E	TONE EA8E
8F	RANDOM COMPUTER
90-A7	TONES EA90-EAA7

Appendix F: Change Log

V1.0 – Initial Release
V1.1 –

- Disclaimer - updated
- Section 1.0 – Corrected processor timing
- Section 2.0 – Correct WR signal description
- Section 3.0 – Changed external RAM size to 128 bytes
- Section 4.6 – Corrected control register bits
- Section 4.7 – Corrected status bits functions
- Section 4.8 – Corrected collision register
- Added Section 4.10 – sound
- Section 4.12 - Removed
- Section 4.14 – Altered to reflect the true function of register $A5
- Section 6.1 – Added remaining cart vectors
- Appendix D – Fix register functions
- Appendix E - Added