This document describes the registers which are accessible through the nec monitor chip in the IS-NITRO devices.

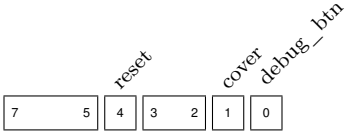
List of Registers

1 0x08000000 Registers

Those registers should be written as 16 bit, but only the lower 8 bits are actually valid and used.

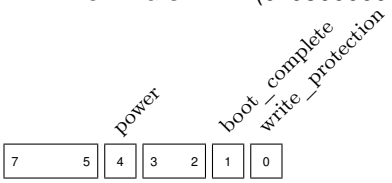
1.1 Nitro Registers

Register 1.1: NITRO REGISTER 0 (0x08000000)



reset	0 = resetting, 1 = normal
cover	1 if the cover of the ds is closed
debug_btn	State of the debug button

Register 1.2: NITRO REGISTER 1 (0x08000002)

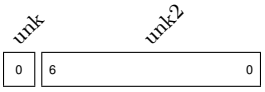


power	Related to power, setting to 1 and then to 0 carries out a reset (powercycle?)
boot_complete	Indicates if boot has been completed. Most likely based on the value of the post boot flag (0x04000300) register which indicates the game has started and debugging may occur. Jtag does not work as long as this flag is not 0.
write_protection	Idk what this does, 0 = on, 1 = off. Might be for rewriting the protected part of the flash memory (requires to short a pad on a normal ds). This flag is in register 0x08000004 on older hardware revisions as it seems.

1.2 Forward Registers

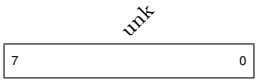
Registers related to capturing footage to pc.

Register 1.3: FORWARD REGISTER 0 (0x08000004)



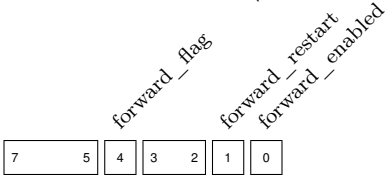
unk	Idk
unk2	Idk. This register was 0x08000010 in older hardware revisions.

Register 1.4: FORWARD REGISTER 1 (0x08000006)



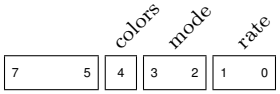
unk	Idk. This register was 0x08000012 in older hardware revisions.

Register 1.5: FORWARD ENABLE (0x08000008)



forward_flag	Idk
forward_restart	1 = restarting?
forward_enabled	Forwarding of video frame is enabled when 1

Register 1.6: FORWARD CONFIG (0x0800000A)

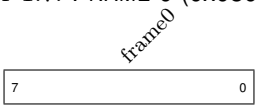


colors Probably if RGB555 or RGB666 should be used, unknown which value is which

mode Forwarding mode?

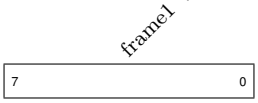
rate Forwarding framerate, most likely one of 15, 20, 30 or 60; but unknown what the values are

Register 1.7: FRAME 0 (0x0800000C)



frame0 Related to adding a frame around the captured images

Register 1.8: FRAME 1 (0x0800000E)



frame1 Related to adding a frame around the captured images

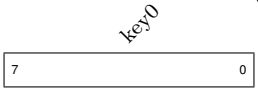
1.3 Monitor Registers

Registers related to analog video output.

1.3.1 Monitor Unlock Registers

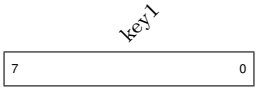
Video output is not enabled until the keys are written into the unlock registers. It seems that writing to these registers after unlocking has no effect.

Register 1.9: UNLOCK REGISTER 0 (0x08000010)



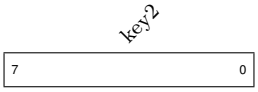
key0 Should be set to 0x59 for unlocking

Register 1.10: UNLOCK REGISTER 1 (0x08000012)



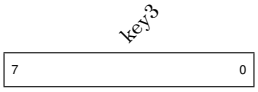
key1 Should be set to 0x4F for unlocking

Register 1.11: UNLOCK REGISTER 2 (0x08000014)



key2 Should be set to 0x4B for unlocking

Register 1.12: UNLOCK REGISTER 3 (0x08000016)

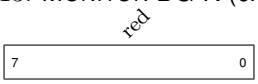


key3 Should be set to 0x4F for unlocking

1.3.2 Monitor BG Registers

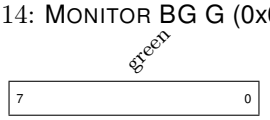
Sets the background color for the video output.

Register 1.13: MONITOR BG R (0x08000018)



red The red component of the monitor bg color.

Register 1.14: MONITOR BG G (0x0800001A)



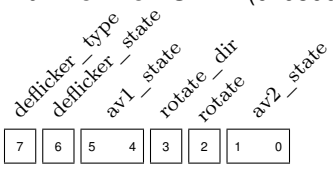
green The green component of the monitor bg color.

Register 1.15: MONITOR BG B (0x0800001C)

blue The blue component of the monitor bg color.

1.3.3 Monitor State Registers

Register 1.16: MONITOR STATE (0x0800001E)

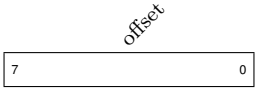


deflicker_type	Type of de-flicker applied? Turning on gives weird results
deflicker_state	1 if de-flicker is enabled
av1_state	Sets what to output on av1. (0 = nothing, 1 = top, 2 = bottom, 3 = both)
rotate_dir	The direction of rotation
rotate	Enables rotation of the 2 screens
av2_state	Sets what to output on av2. (0 = nothing, 1 = top, 2 = bottom, 3 = both)

1.3.4 Monitor Cursor Registers

The following registers configure the display of the current touch position with a cursor. To disable cursor display set both x and y to 0xFF

Register 1.17: CURSOR IMAGE OFFSET (0x08000022)

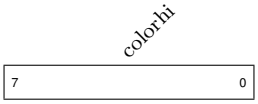


offset Sets the pixel currently selected

Register 1.18: CURSOR IMAGE PIXEL LO (0x08000024)

colorlo Bottom 8 bits of the color of the selected cursor pixel

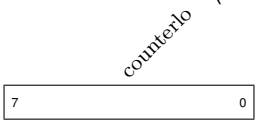
Register 1.19: CURSOR IMAGE PIXEL HI (0x08000026)



colorhi Top 8 bits of the color of the selected cursor pixel. The top most bit is alpha

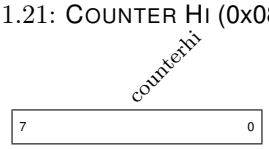
I have no idea if this thing is a real counter or some rapidly changing value. It seems to go up though.

Register 1.20: COUNTER LO (0x08000028)



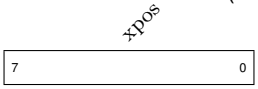
counterlo The lower 8 bits of the counter

Register 1.21: COUNTER HI (0x0800002A)



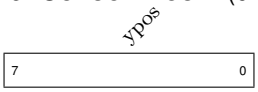
counterhi The upper 8 bits of the counter

Register 1.22: CURSOR POS X (0x0800002C)



xpos The x position of the cursor (in screen coordinates)

Register 1.23: CURSOR POS Y (0x0800002E)

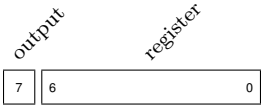


ypos The y position of the cursor (in screen coordinates)

1.3.5 Monitor Config Registers

Configures video output. For example the display position and if the output is interlaced can be configured. The following registers are only valid when writing.

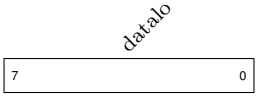
Register 1.24: MONITOR REGISTER SELECT (0x08000030)



output The output of which the register is selected

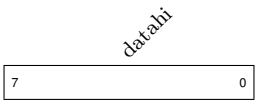
register Sets the currently selected register (0 = ?, 1 = ?, 2 = ?, 3 = ?, 4 = ?, 5 = interlaced)

Register 1.25: MONITOR REGISTER DATA LO (0x08000034)



datalo Bottom 8 bits of the value in the selected register

Register 1.26: MONITOR REGISTER DATA HI (0x08000036)



datahi Upper 8 bits of the value in the selected register

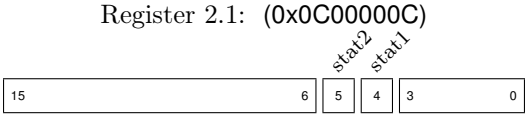
2 USB Registers (0x0C000000)

0x0C000002 ?; values written: 0

0x0C000004 ?; values written: 0, 0x10

0x0C00000C When bits 4 and 5 are 0, packet sending has finished as it seems

Register 2.1: (0x0C00000C)

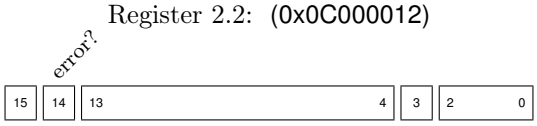


stat2

stat1

0x0C000012

Register 2.2: (0x0C000012)



error? Seems to signal an error

0x0C000014

0x0C000016

0x0C000018

0x0C00001E ?; values written: 0x3FFF

0x0C000020

0x0C000022 ?; values written: 0x3FFF

0x0C000026

0x0C000028 ?; values written: 0, 2

0x0C00002C ?; values written: 0xC, 0x30

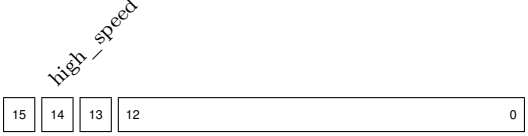
0x0C00002E ?; values written: 2

0x0C000030 ?; values written: 0

0x0C000034 ?; values written: 0x100

0x0C000036

Register 2.3: (0x0C000036)



high_speed 1 if the usb interface is running in high speed (usb 2.0) mode

0x0C000046

0x0C000048 EP0 Control IN (to pc) FIFO (16 bit units)

0x0C00004A EP0 Control IN (to pc) FIFO (8 bit units)

0x0C00004E EP1 Bulk OUT (from pc) FIFO (16 bit units)

0x0C000050 EP1 Bulk OUT (from pc) FIFO (8 bit units)

0x0C000052 EP1 Bulk OUT (from pc) FIFO Data Count

0x0C000056 EP2 Bulk IN (to pc) FIFO (8 bit units)?

0x0C000058 EP2 Bulk IN (to pc) FIFO (16 bit units)

0x0C000080 ?; values written: 1

0x0C000082 ?; values written: 0

0x0C000084 ?; values written: 0, 1

0x0C000086 ?; values written: 0

0x0C000088 ?; values written: 0

0x0C00008E ?; values written: 0

0x0C0000A0 ?; values written: 0x22

0x0C0000A4 ?; values written: 0, 1

3 Sub LCA (0x0C800000)

0x0C800000

0x0C800010

0x0C800012 Sub LCA Hardware Version

0x0C80001C

4 AGB Bus (0x0F800000)

0x0F800000 ADSRAM? 256kb debug memory mapped to the gba slot

0x0F840000 Main LCA1 Hardware Version

0x0F840004

0x0F840006

0x0F841000 ADSRAM offset; the offset in agb memory where the 256 kb of debug memory is mapped. Encoded as (dsaddress - 0x08000000) » 18

0x0F841008

0x0F84100A

0x0F842000

0x0F84200C

0x0F843000 AGB bus protection bits. 512 bytes of which each bit sets if a 8192 byte region of the gba slot is accessible (1 = accessible). The normal protected config is to set the first 256 bytes to 0, except for the first which is set to 1 and the second 256 bytes to 0xFF. This yields a protected region from 0x08002000 to 0x09000000.

0x0F844000

0x0F844002 JTAG state (enable/busy and error bits as it seems)

0x0F844004 JTAG bit count

0x0F844008

0x0F84400A

0x0F84400C

0x0F845000 JTAG tdi stream

0x0F846000 JTAG tms stream

0x0F847000 JTAG tdo stream

0x0F860000

5 NDS Bus (0x0FC00000)

0x0FC00000 Part of the nitro card emulator memory space?

0x0FC40000 Top 16 bits of the mapped emulator memory?

0x0FC40010

0x0FC40012

0x0FC40018

0x0FC40028

0x0FC4002A

0x0FC40030 Main LCA2 Hardware Version

0x0FC40032

0x0FC40036

0x0FC40040

0x0FC40042

0x0FC40200 256 bytes of something?

0x0FC40300 256 bytes of something?