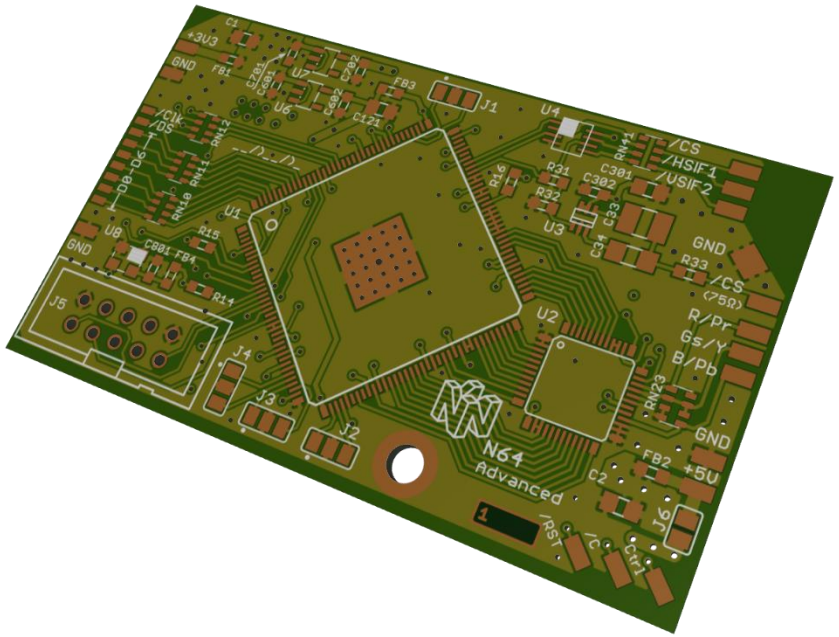
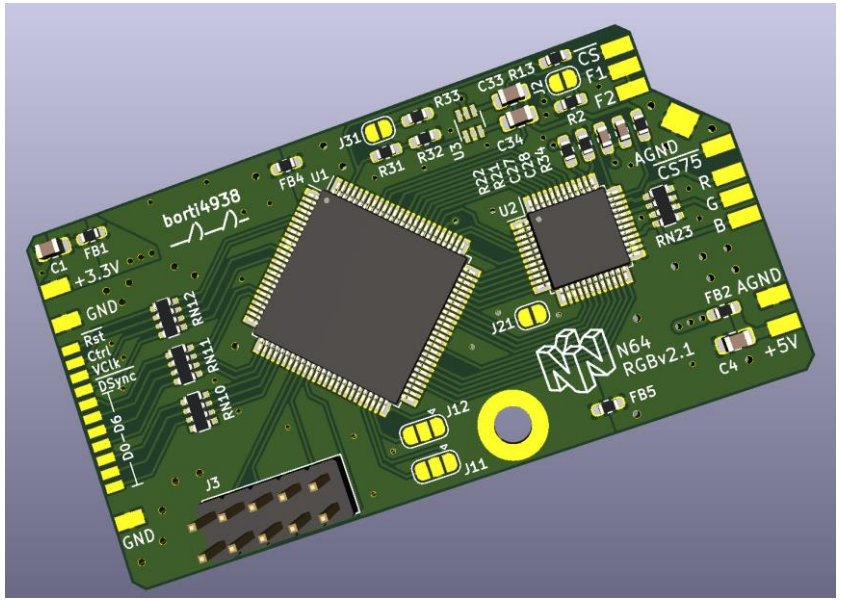
N64 Advanced
and
N64RGB Version 2.1

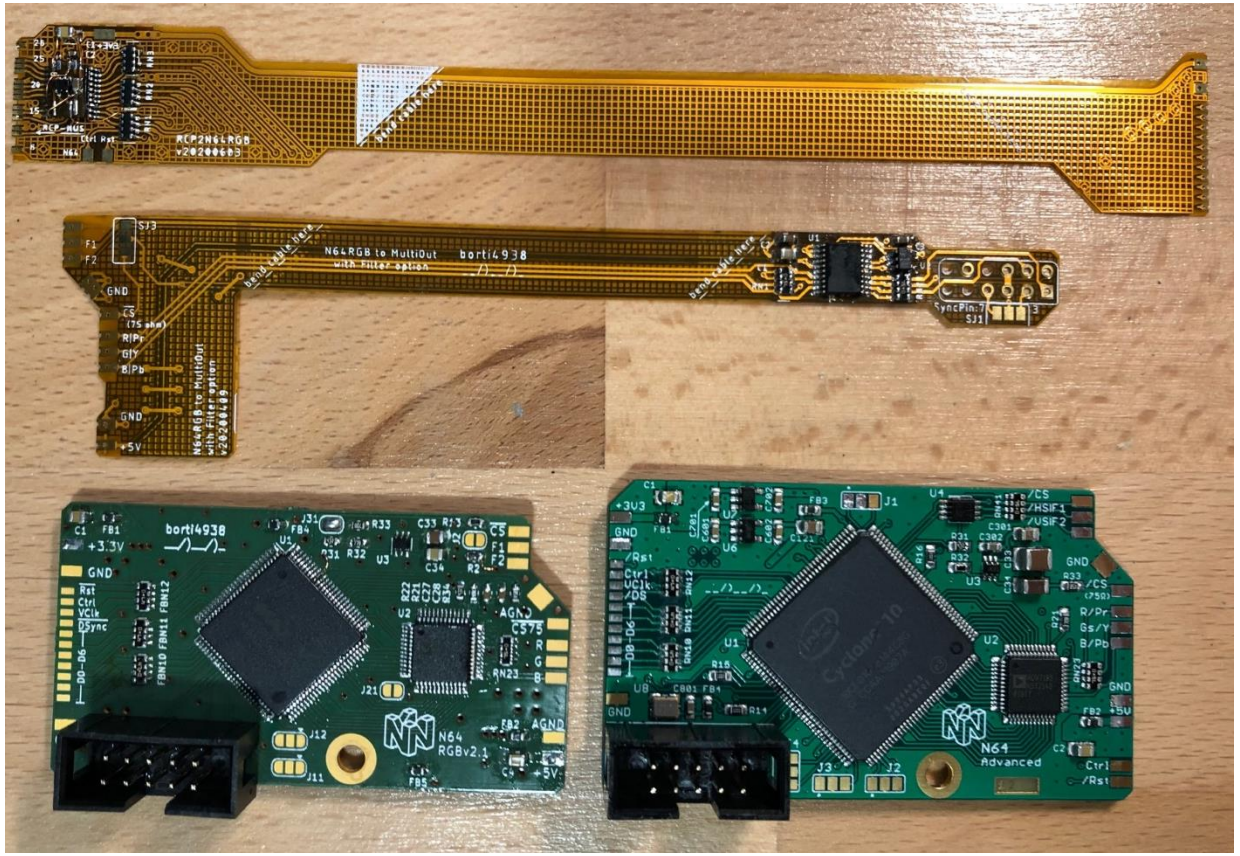


Installation and
Setup Guide



Content

- Open the Console
- Preparation
- Solder Work
  - Console
  - Modding Kit
- Settings (Solder Jumper)
- Cable SetupController
  - In Game Routines (all versions)
- Menu (N64A only feature)
- Firmwareupdate



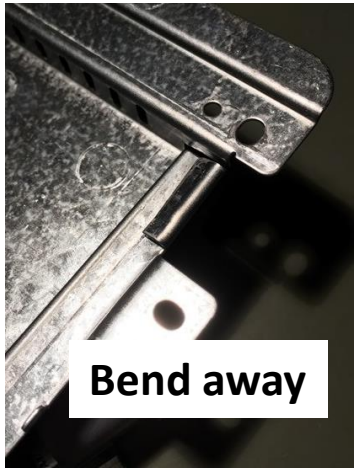
(pictures often show prototypes or earlier PCB versions)

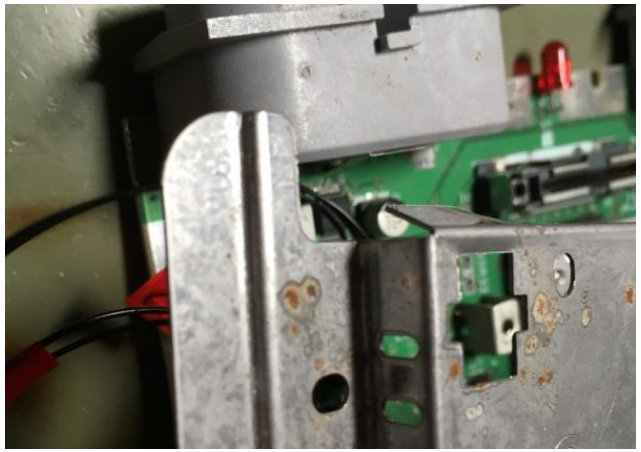
Open the Console

- Remove Jumper Pak / Expansion Pak
- Remove screws from bottom side of the console (needs 4.5mm game bit tool)
- Lift up the top housing
- Remove marked screws from inside (in some later consoles heat sink is designed differently)
- Pull out the mainboard and remove heat sink and RF shield
- **Hint:** Now you have a good chance to clean up your N64 shell under hot water.

Picture by Zerberus (circuit-board.de user)

Preparation – Top RF shield

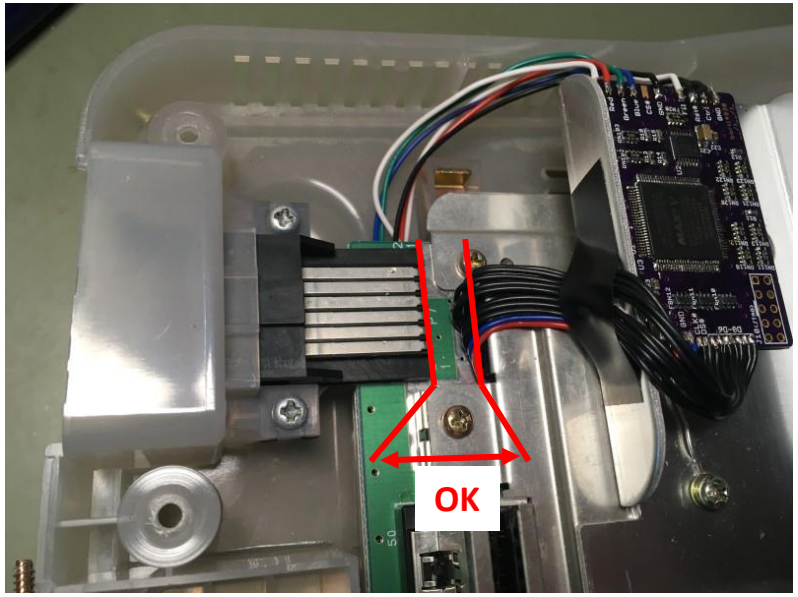


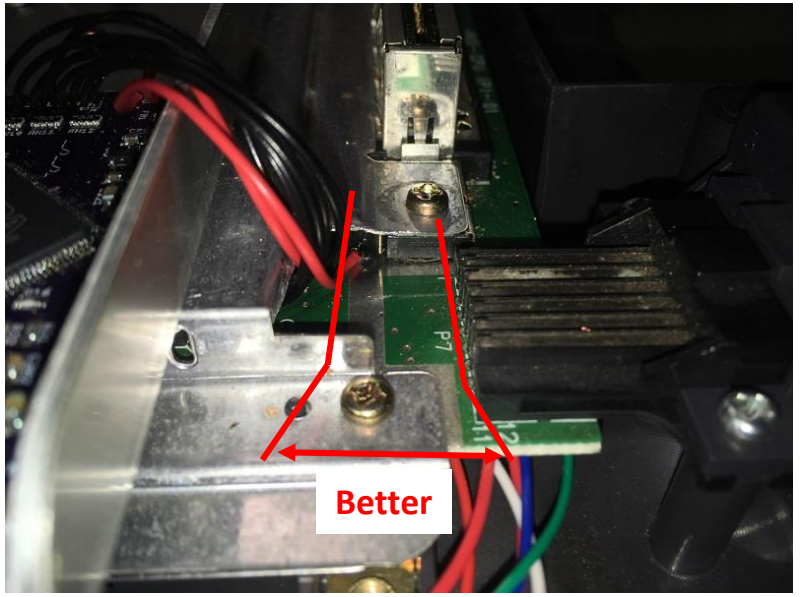


- Next to the MultiOut and the cartridge slot
  - Locate the tap in the RF shield closing the gap to the N64 mainboard
  - Bend away or remove this tap or even cut off a small piece out of the RF shield as shown here
- At the front side between reset button and controller port:
  - Slightly bend off the tap here
- This helps later to fit the connection wire

Preparation – Top RF shield

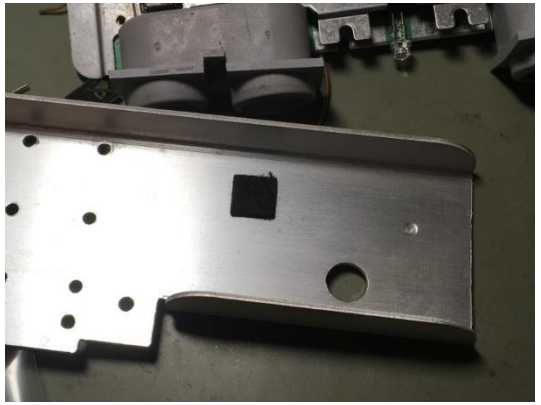
-- Interference issue--





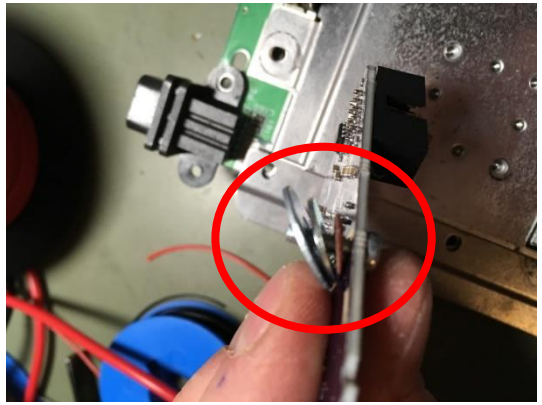
- Later, the data lines as well as the clock will be routed close to the MultiAV
- The distance between both is a crucial issue as your wires act as interferer
- Hence, have tap bend might be ok, but it is better to cut some piece of the RF shield away to further increase the distance between MultiAV and data lines

Preparation – Mount the PCB



- Put a small piece of a flat isolationg foam or glider under the top left of the PCB (OPTIONAL)

- Apply some amount of solder to the mounting hole
- This helps later to have a better connection to the screw



Mounting material:

- Screw
  (M3 diameter,
  min. 10mm, max. 14mm)
- 3x washer below PCB:
  - isolating or small outer diameter
    (three traces are close to the mounting hole)
  - a slightly larger washer
  - a large washer
    (such that this washer does not fit through the mounting hole of the heat sink)

Preparation – Mount the PCB

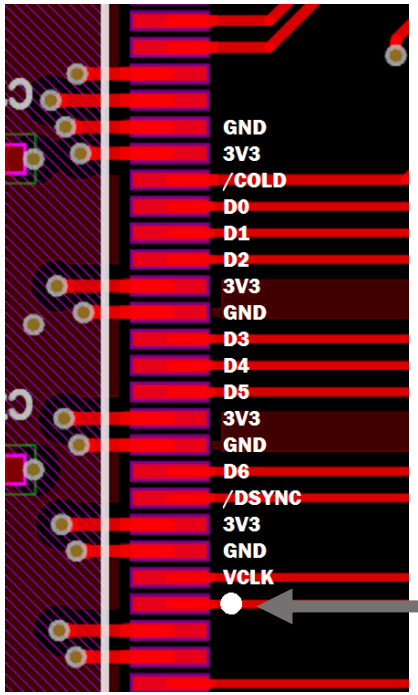
- Place the PCB on top of the heat sink

- Fasten screw from the bottom side of the heat sink
- Use another washer and a (locking) nut
- **Attention:**
  - Do not place the washer into the RF shield area!
  - Do not stress the PCB to much!

Solder Work – Console

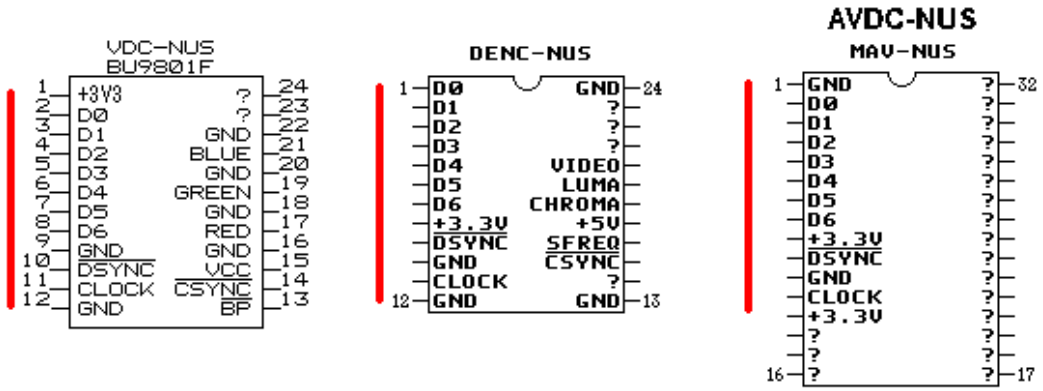
-- digital video signals --

Most of the signals needed has to be taken from video processor output; the **RCP-NUS**.
Next to the RCP-NUS is the video encoder, where several types are used during the variety of N64 designs.
The pinouts are given below. Basically we need D0-D6, /DSYNC and /CLK (or VCLK or CLOCK) for the modding board.



Pin 10 (marked with a dot)

Pinout RCP-NUS

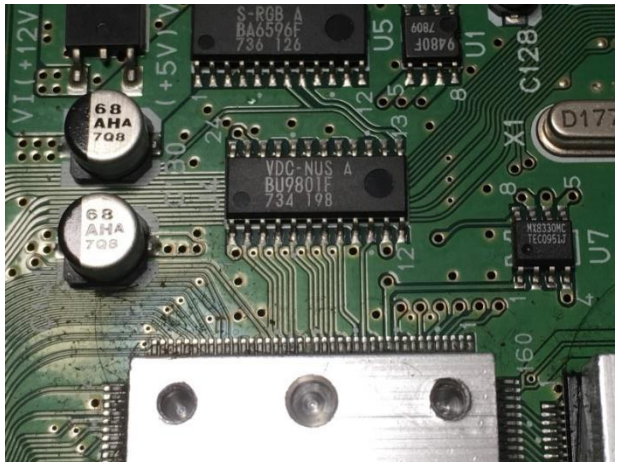


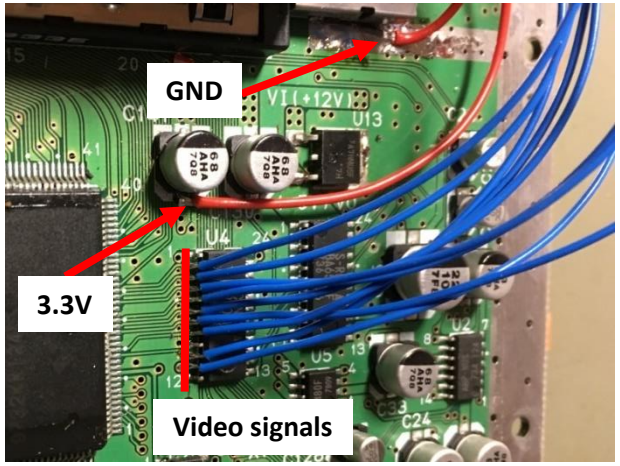
Pinouts of different encoder chips used in the N64

RCP-NUS picture by Marshall
Encoder picture by Viletim

Solder Work – Console

-- digital video signals and 3.3V and GND --

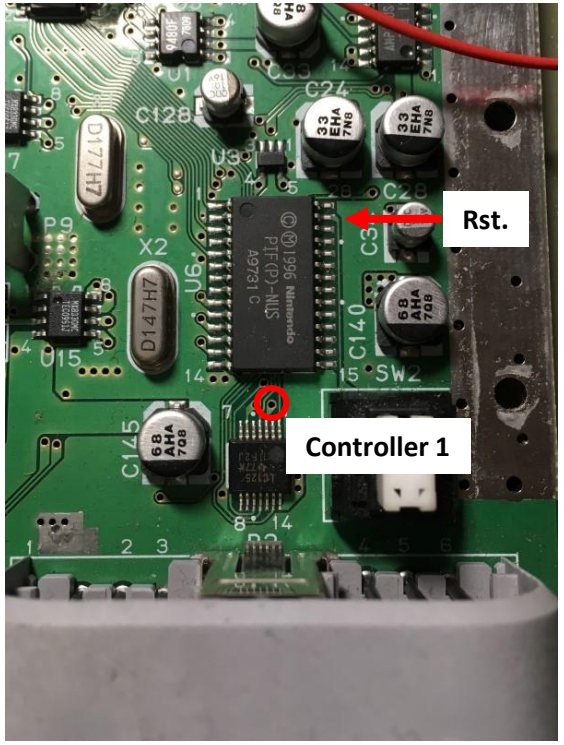


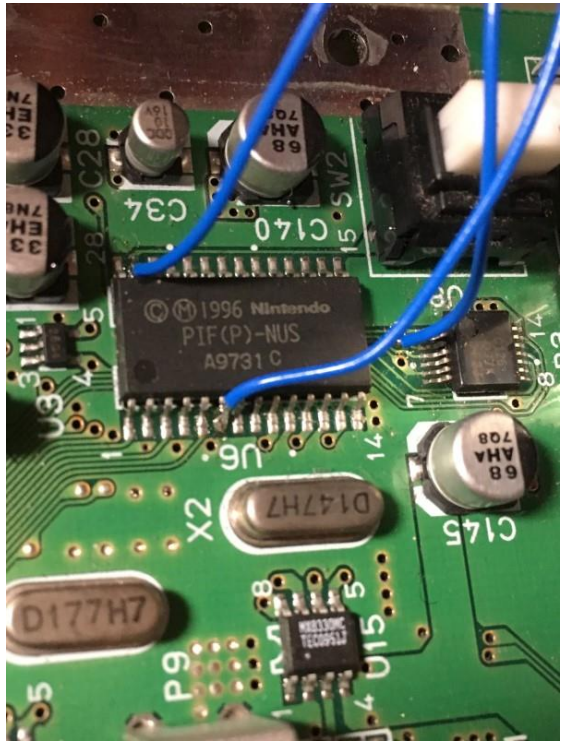


- Locate the video encode (here a VDC-NUS)
- Solder a bunch of wires to the video signal pins
  - You may use an adapter if you have a MAV-NUS or AVDC-NUS encoder (see README)
- Solder two addional wires
  one for 3.3V (can be picked off C141) and
  one for GND (large GND plain)
- Route all wires approximately to the MultiAV port

Solder Work – Console

-- Controller and Reset --

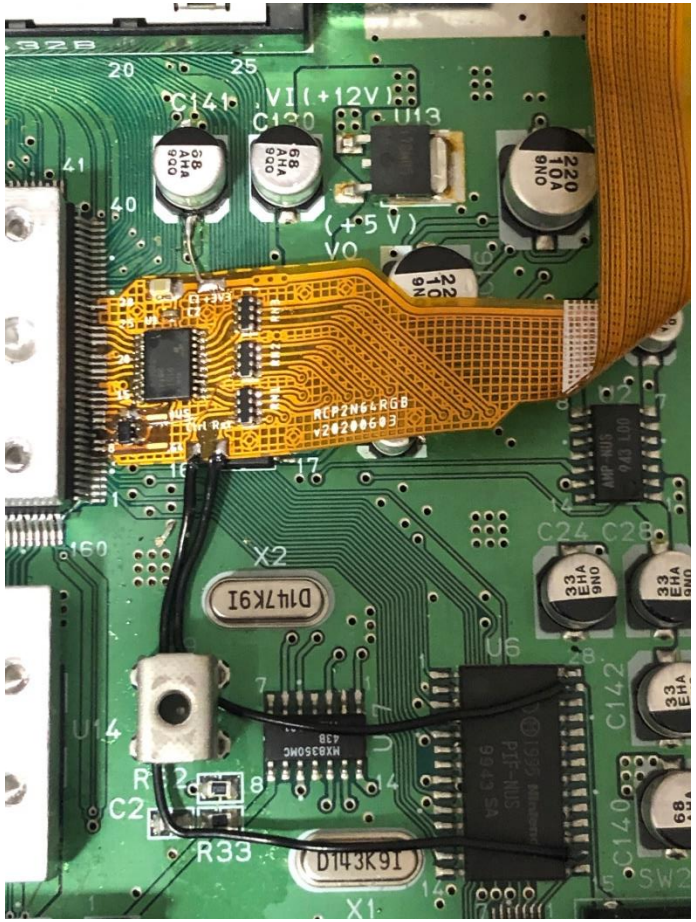




- All signals for the controller and the reset can be picked off the PIF-NUS which is located next to the reset button
- Solder the following wires:
  - One for reset (PIF-NUS pin 27)
  - One for the controller input
    in my case: the marked via, sometimes also PIF-NUS pin 16
    in any case: search for a suiteable connection to the middle pin of controller port 1

Solder Work – Console

-- Using the flex cable --

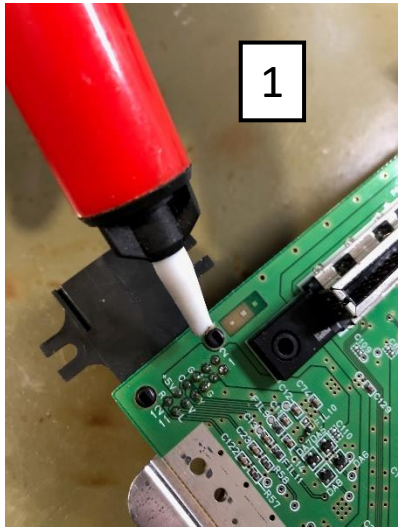


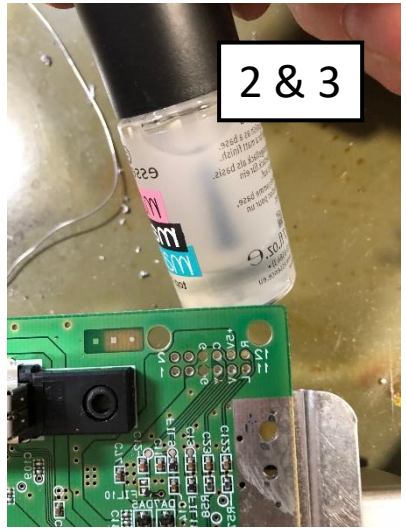
- If using the flex cable, solder the RCP connector side to the RCP-NUS as shown
  - first pin bottom – 8
  - last pin to – 28
- Connect 3.3V to the flex
  - e.g. taken from C141
- Connect Ctrl. and reset
  - reset from PIF-NUS pin 27
  - Controller from PIF-NUS pin 16
    - Make sure that PIF-NUS pin 16 is connected to the middle pin of controller port 1, otherwise search for a suiteable connection point
- Note: when using the flex you need to cut the RF shield near the MultiOut as shown a few pages above

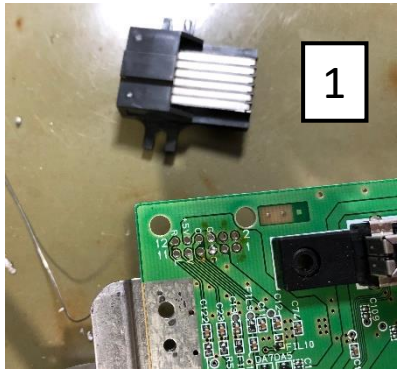
Solder Work – Console

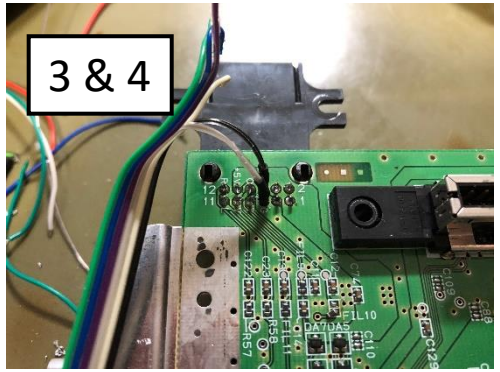
-- MultiAV port – GND insulation --

- <u>This step is</u>
  - **Recommended** for the N64RGBv1
  - **Optional** for the N64RGBv2 and N64Advanced, yet from the electrical point of view recommended
- **<u>Insulate the GND from the MultiOut:</u>**
  1. **Desolder** the MultiOut connector
  2. **Drill** out the GND pads
  3. **insulate** the drills (either with varnish or small shrink tubes) in case of tiny copper remains
  4. **Solder** MultiOut back into place
- <u>Why is it needed?</u>
  - By placing the modding board rather far away from the mainboard the installation creates a so called GND loop
  - If the GND loop is too long, some current (switching current) will flow to GND over signal wires, i.e. also over the video outputs which will be visible in distortions / noise
  - This is very dominant on the N64RGBv1 as the digital and analog circuit are supplied by 3.3V.
    On the N64RGBv2 and N64Adv, the digital and analog part are supplied with 3.3V and 5V, respectively. So the effect is neclectable to my experience.



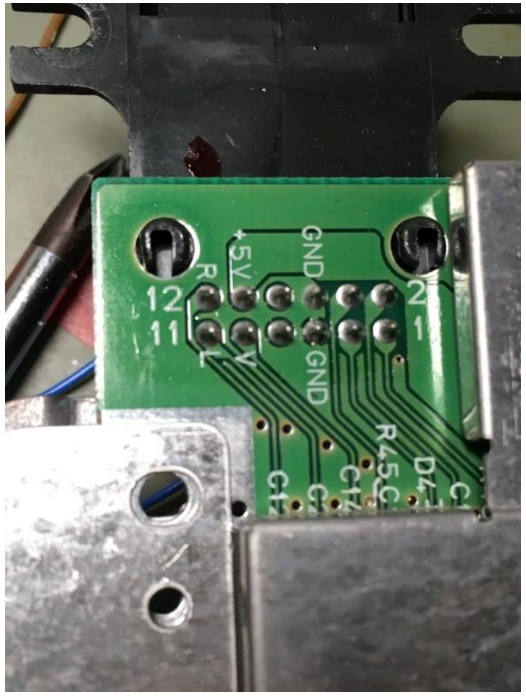


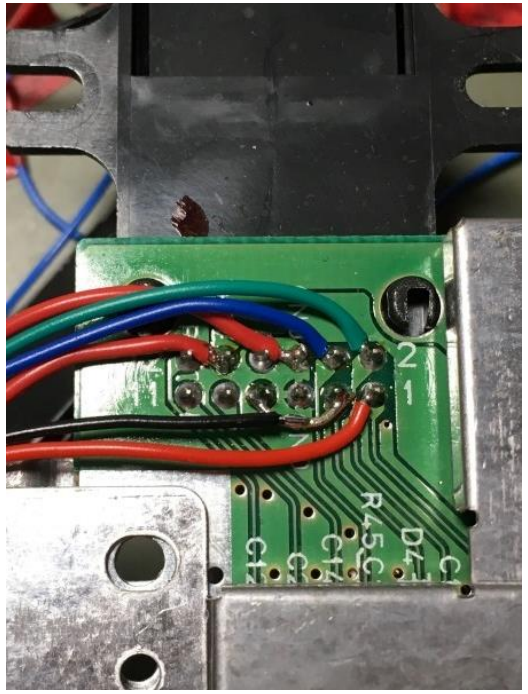




Solder Work – Console

-- MultiAV port – analoge video output --



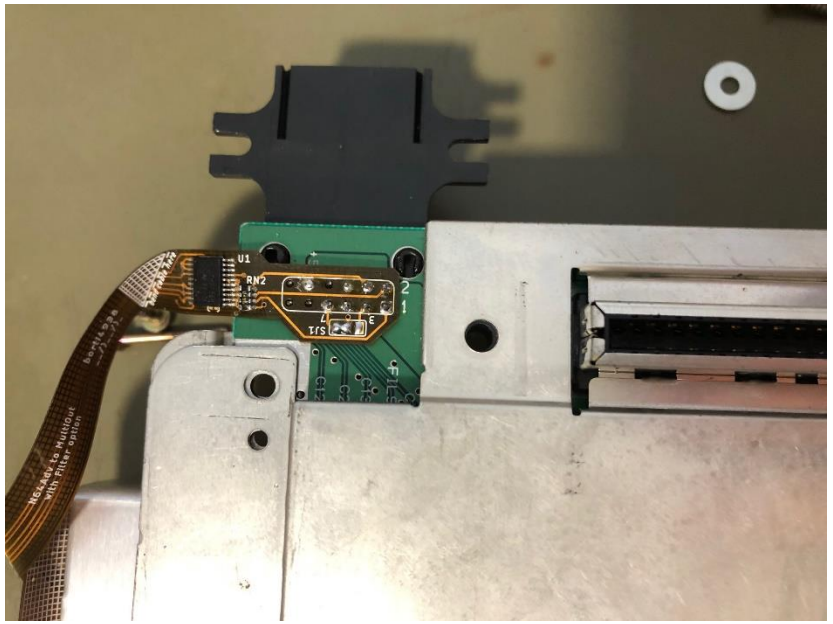


- Prepare some wires for:
  - Pin 1: Red / Pr
  - Pin 2: Green / Y
  - Pin 3: Sync *
  - Pin 4: Blue / Pb
  - Pin 5 / 6: GND
  - Pin 7: Sync *
  - Pin 10: +5V

*** Sync:**
You need to take the composite sync out of the modding board. Depending on your RGB cable, you need to use pin 3 (sync on csync cable) or pin 7 (sync on luma cable). I soldered a bridge between both pins to make it useable with both types of cables.
Sometimes pin 3 / 7 is used in your stock console. **Make sure that the pin / these pins is / are unconnected.**

Solder Work – Console

-- MultiAV port – analoge video output with flex cable--

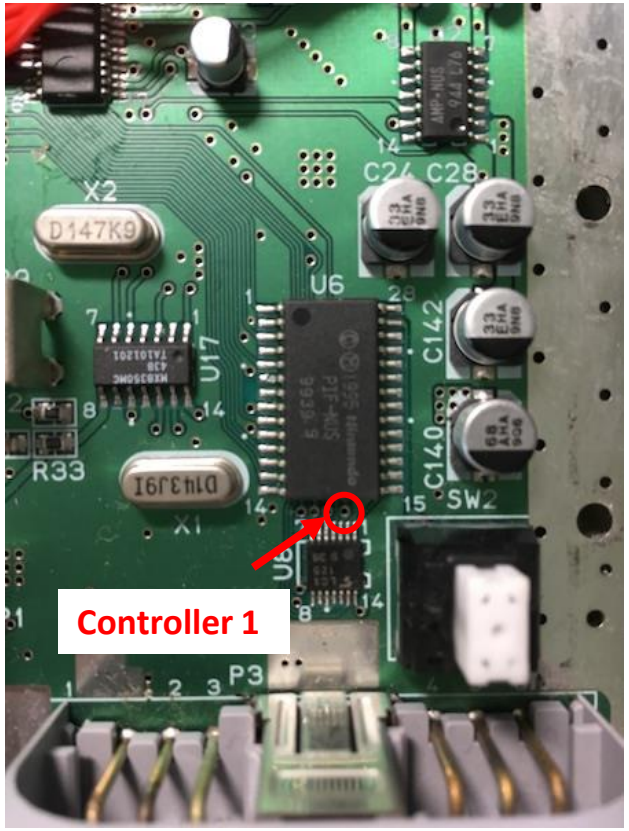


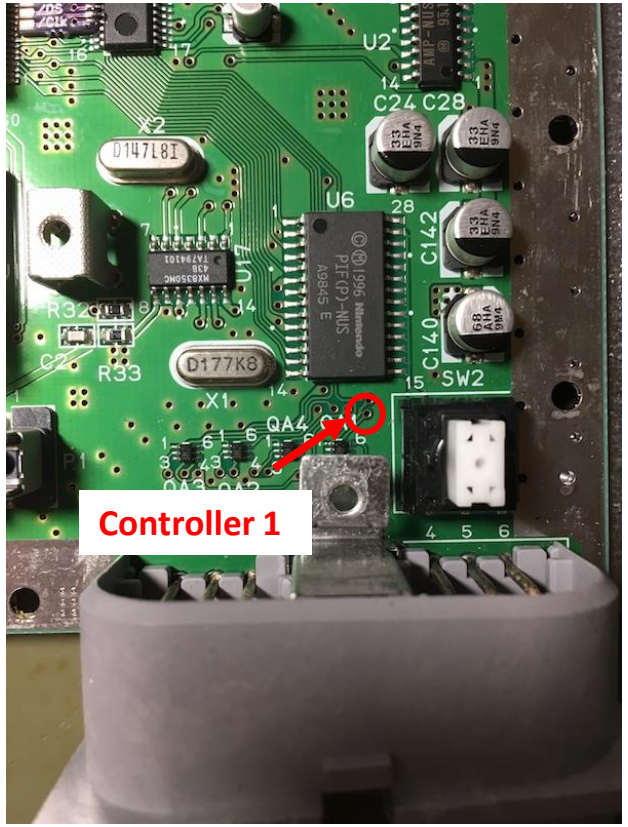
- Simply connect the flex cable as shown with the MultiOut
- Make sure that you prepared the GND as presented two pages above, ideally

*** Sync:**
You need to take the composite sync out of the modding board. Depending on your RGB cable, you need to use pin 3 (sync on csync cable) or pin 7 (sync on luma cable). Use SJ1 to determine this. You may close both sides of SJ1. Sometimes pin 3 / 7 is used in your stock console. **Make sure that the pin / these pins is / are unconnected.**

Solder Work – Miscellaneous

-- Other PIF-NUS configurations and proper Controller 1 solder points --





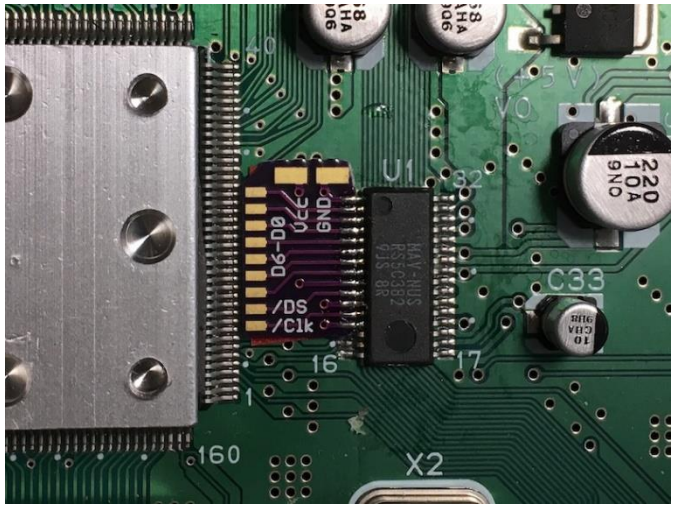
Solder Work – Miscellaneous

-- Break Out Adapter --

RCP2Pads

(no picture atm)

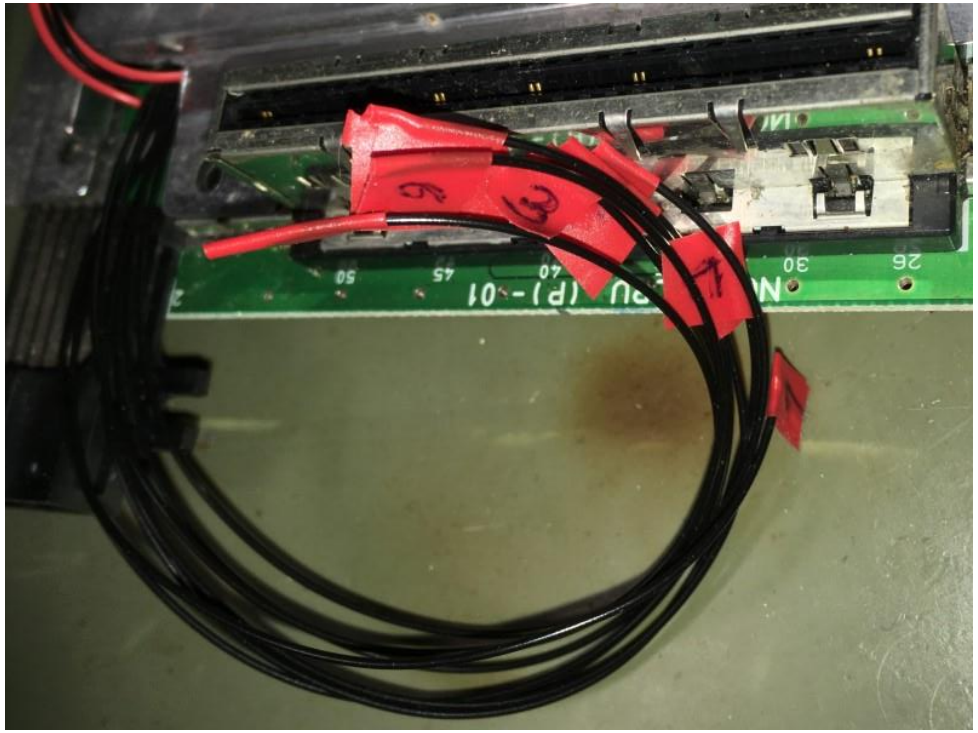
- RCP2Pads flex breakout
  - This flex is a small version of the RCP2N64RGB flex cable (e.g. use this to limit costs)
  - Connect the RCP2Pads in the same way as the RCP2N64RGB and use wires to connect the flexible break out board to the modding board.



- MAV-NUS break out (outdated)
  - The MAV-NUS and the AVDC-NUS have a small pitch of 0.8mm
  - You may want to use the MAV-NUS breakout board to work on a 1.27mm pitch
  - Check connectivity between the RCP-NUS and the pads on the breakout with a DMM
  - Check for adjacent shorts

Solder Work – Miscellaneous

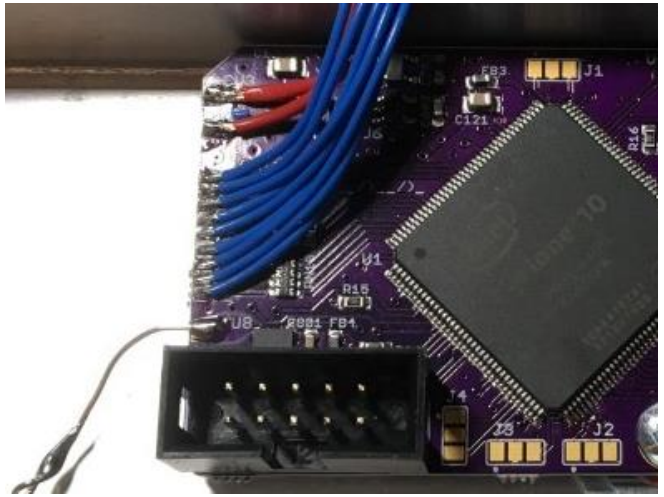
-- Marking wires --

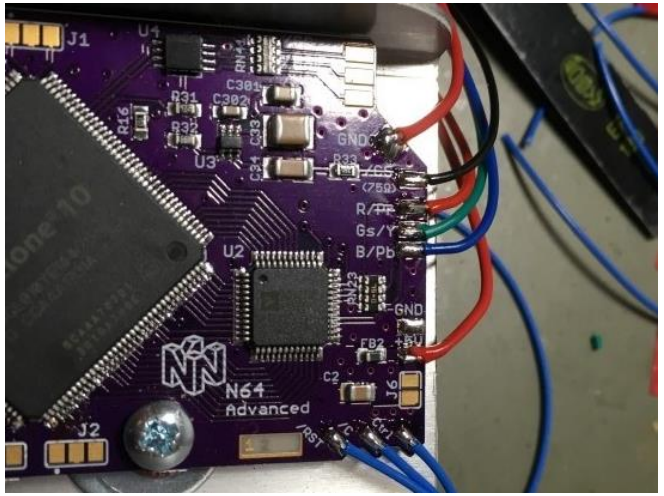


- If you use lose wires as I do, you may mark them before you go on

Solder Work – Modding Kit

-- With wires --

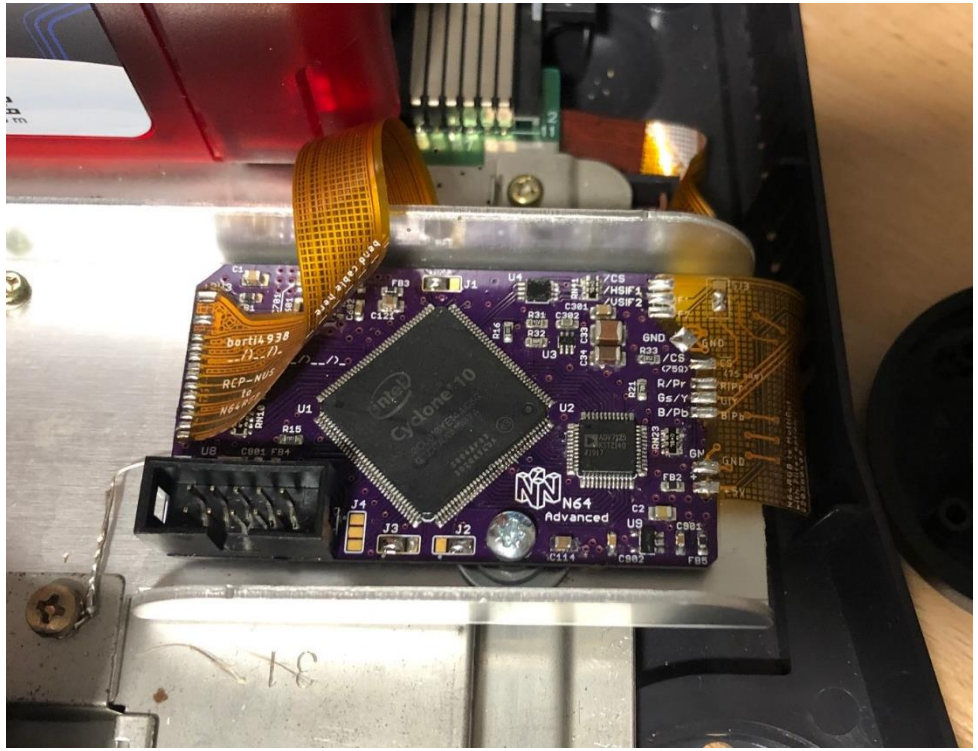




- Reassemble the top RF shield and heat sink on top of the N64 mainboard
  - Make sure that you do not squeeze any of the wires you prepared
  - For that you cut and/or bend the taps of the RF shield
- Connect +3.3V and GND to the left hand side of the modding kit
- Connect /Rst., Ctrl. (Controller), the seven data lines and /DSYNC and VCLK to the appropriate pads
- Connect the video lines, GND and +5V to the right hand side of the modding kit
- set jumpers to finish your installation

Solder Work – Modding Kit

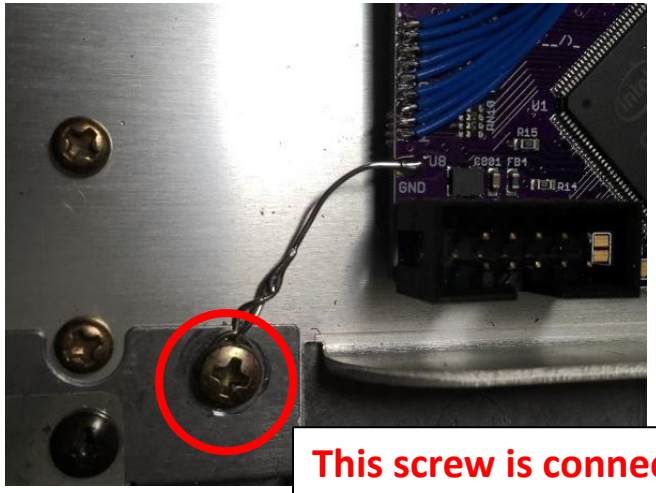
-- With Flexible Cables --

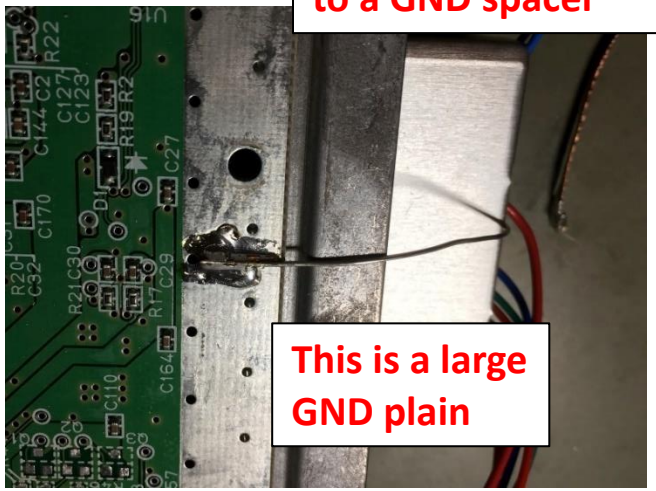


- With the full flex cables, you simply have to solder flex cables to the modding board
- Some hints:
  - Gently bend the cables as marked on silk screen
  - Check for adjacents shorts
- set jumpers to finish your installation

Solder Work – Modding Kit

-- Some extra GND --

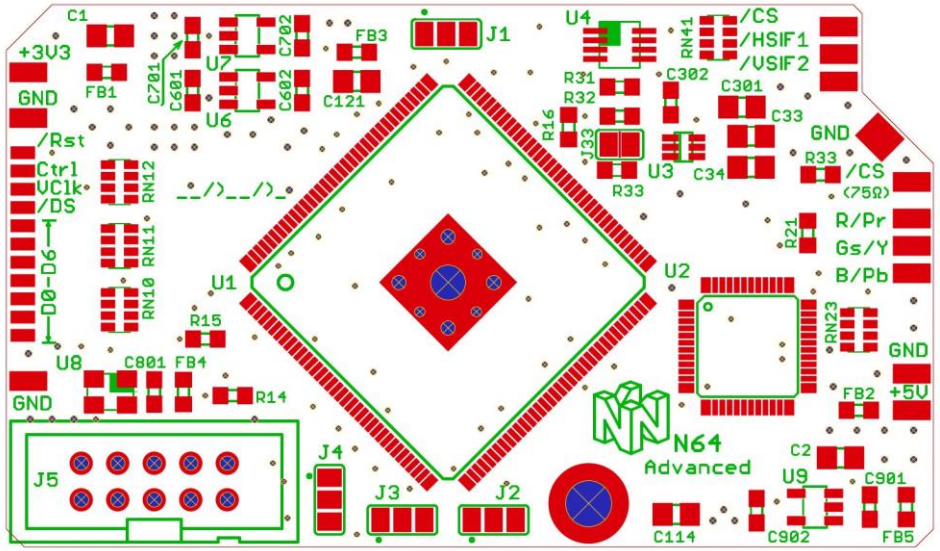




- To reduce the GND loop size, it is helpful to assemble some extra GND connections to the N64 mainboard
- If the GND is not properly designed between the N64 mainboard and modding kit some current out of the digital signals flows back to the N64 throgh the analog video connections. This causes visible noise!
- I assembled two extra connection: one on the left hand side of the modding PCB and one at the right hand side.

Settings – Solder Jumper (N64 Advanced)

J5: 10pin JTAG interface
for update firmware



J4: Linemode
J4.1: - opened: linedoubling enabled
- closed: no linedoubling
(beats J4.2)
J4.2: - opened: 480i pass through
- closed: 480i de-interlace (bob)

J1: VGA sync / Filter AddOn
J1.1: - opened: output HS, VS and CS
(VGA output possible)
- closed: use filter addon
J1.2: - Use filter board in bypass mode
(e.g. to use it as a simple
NTSC output to PAL output conversion)

J33: C-Sync Level @ 75ohm termination
(/CS (75ohm) pad)
- opened: 1.87Vpp
- closed: 300mVpp (recommended)

Note:
Some jumpers (J2, J3 and J4) are outdated since the menu driven fw. is up. However, J1.1 must be set if the filter board is used. The remaining jumpers are used to set defaults if the configuration cannot be loaded from flash (e.g. after a fw. update)

J3: Scanline strenght
(J3.2/J3.1) =
- (opened/opened): 0%
- (opened/closed): 25%
- (closed/opened): 50%
- (closed / closed): 100%

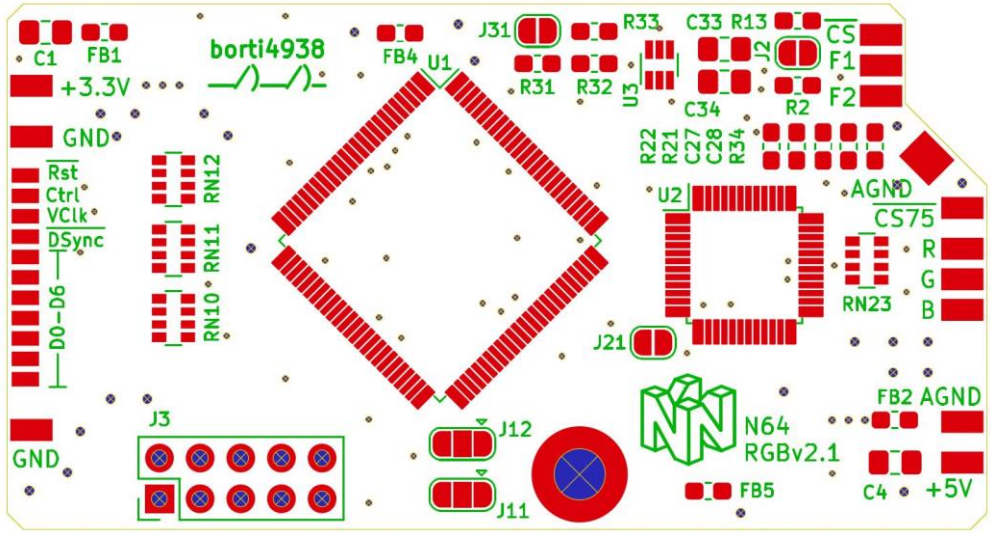
J2: RGB, RGsB, YPbPr
J2.1: - opened: RGB output
- closed: RGsB output
J2.2: - opened: RGB / RGsB output
- closed: YPbPr output (beats J2.1)

Notes for dual jumper:
- Jx.1 is always marked with a dot
- Closed reference is middle pad

Settings – Solder Jumper (N64 RGBv2.1)

J11: VI-DeBlur and 15bit mode
J11.1: - opened: vi-deblur off
- closed: vi-deblur on
J11.2: - opened: 15bit mode off
- closed: 15bit mode on
(J11 acts as power-cycle default)

J12: IGR function control
J12.1: VI-DeBlur and 15bit mode
- opened: ctrl. functions on
- closed: ctrl. functions off
J12.2: Reset by controller
- opened: ctrl. function on
- closed: ctrl. function off



Notes for dual jumper J1#:
- J1#.1 is marked with an arrow
- Closed reference is middle pad

J2: Filtersetting (for Filter AddOn)
- opened: Use LPF
- closed: Bypass filter

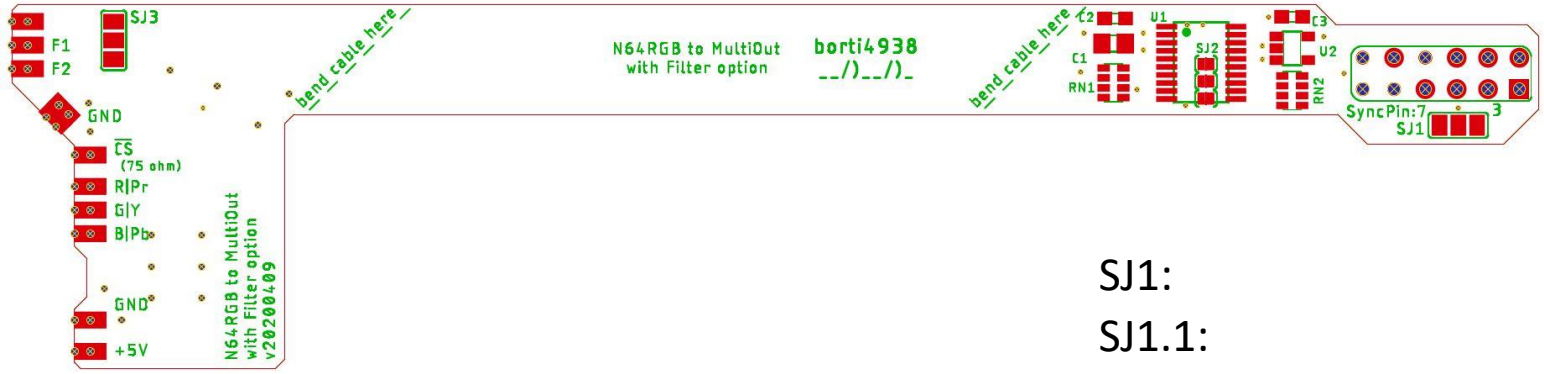
J31: C-Sync Level @ 75ohm termination (/CS (75ohm) pad)
- opened: 1.87Vpp
- closed: 300mVpp (recommended)

J21: Sync on Green
- opened: Don‘t use sync on green
- closed: Output sync on green

J3: 10pin JTAG interface for update firmware

Settings – Solder Jumper (N64 RGBv2.1)

SJ3: Sync source at the modding board
- J3.1: Use TTL C-Sync
- J3.2: Use 75ohm Sync
- Never use both!



SJ2: flex setup
- close all three jumper if you do not use the filter assembly

SJ1:

SJ1.1:
- closed: forward sync to pin 7 of the MultiOut

SJ1.2:
- closed: forward sync to pin 3 of the MultiOut

Notes for TTL sync:
- Sync signal comes from a 5V line driver with 75ohm output.
- In order to attenuate this for 75ohm termination, you have to have a 225ohm – 1.1kohm resistor in your cable

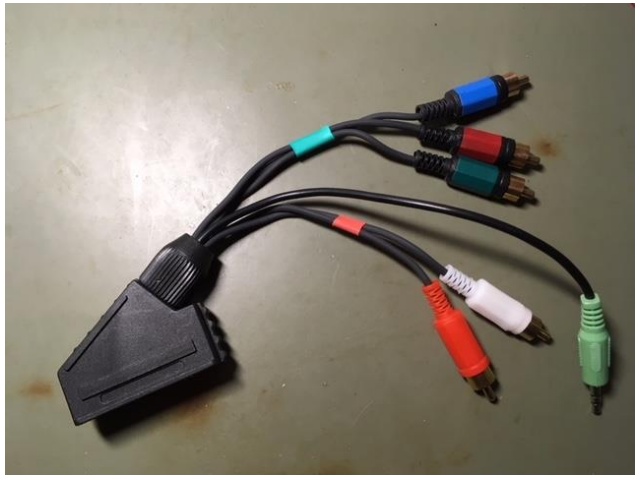
Notes for dual jumper SJ1 and SJ3:
- J#.1 is where the designator is
- Closed reference is middle pad

Cable Setup

-- N64RGBv2.1 and N64 Advanced --

Name	MultiAV	SCART plug	Ref. GND in SCART	Cinch plugs	Notes
Red / Pr	1	15	13	Red plug	Using a 220uF cap in series is possible
Green / Y	2	11	9	Green plug	Using a 220uF cap in series is possible
Blue / Pb	4	7	5	Blue plug	Using a 220uF cap in series is possible
Sync	3 or 7	20	17	Not needed	Consider notes on next page
GND	5, 6	4, 5, 9, 13, 17, 18, 21		To outer ring of each plug	Pin 21 @ SCART plug is outer shield
+5V	10	16	18	Not needed	Use a 180ohm resistor in series
Audio left	11	6	4	Red plug	
Audio right	12	2	4	White plug	

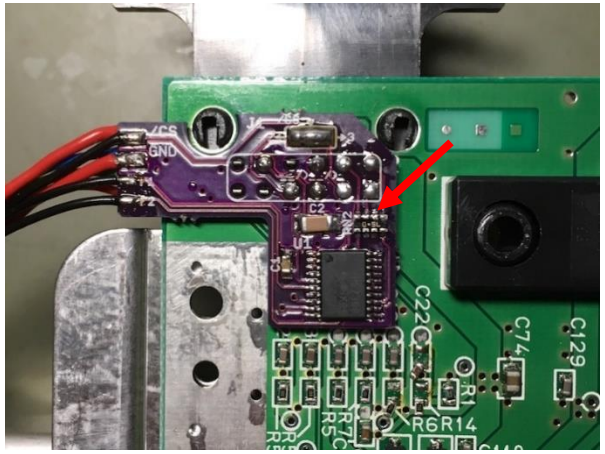
Cable Setup

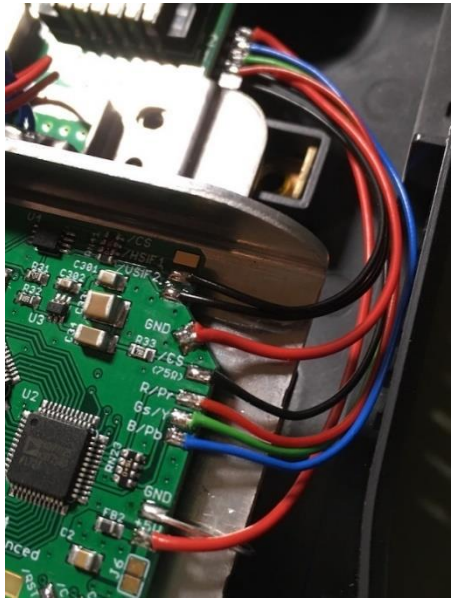


- Notes on Sync:
  - **I recommend using the 75ohm compatible csync output**.
  - If you do so then run a straight wire through your cable for sync.
  - If you use TTL sync output, you have to add an additional resistor in series (N64RGv2: 0ohm to 900ohm; N64A: 270ohm to 1.1kohm) to attenuate the signal for 75ohm terminationed destinations.
- RGB cables:
  - If you buy an RGB cable, buy a typical RGB cable for a NTSC SNES with sync on luma (MultiAV pin 7) or sync on csync (MultiAV pin 3).
  - If you bought a raw csync cable and if you use the 75ohm output, remove / short the resistor which is possibly installed in the sync trace.
- RGsB / YPbPr Cables:
  - With the given table you should be able to build your own component cable or to build an adapter (see pictures on left side)
  - Unfortunately, there are no cables / adapter to bought.

Cable Setup

-- Filter Add On for the N64RGBv2.1 and N64 Advanced --





- Use the Filter AddOn if your ADC device does not apply input filtering (e.g. some capture cards)
- Solder the board onto the MultiAV and decide where /CS has to go – pin 3 and/or pin 7 – depending on your setup
- All connections to the N64RGBv2 and N64A are labeled
- A note on PAL SNES RGB cables:
  - By default the N64RGBv2 and N64A are not suitable to use it with PAL SNES RGB cables
  - By using the Filter AddOn board one can convert the output such that it is possible to buy a stock PAL SNES RGB cable (with sync on luma)
  - Just use the Filter Addon board and use a 39Ohm resistor array at the output (RN2, marked with a red arrow; on the flex cable, it‘s also RN2)
- N64RGBv2.1
  - Use J2 to use or to bypass the filter
- N64 Advanced:
  - Remind to set J1.1 on the N64A
  - Set J1.2 if you want to have the filter off, e.g. in the case you simply want to use a PAL SNES RGB cable
    (actually they will be set to 95MHz cut-off, which is way above the video content)

Controller – In Game Routines

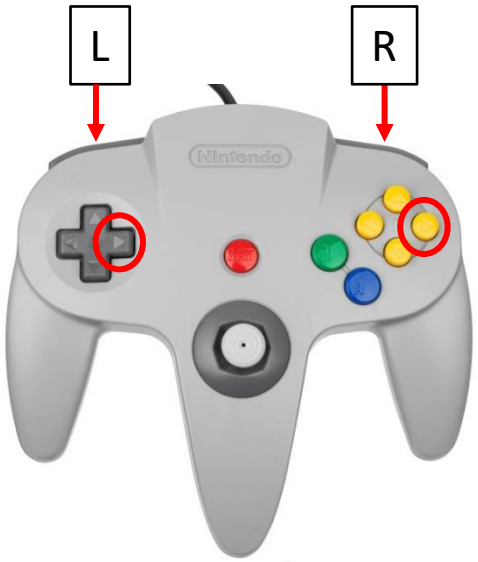
Basis for all button combinations

- The controller is able to trigger a reset or to change de-blur and 15bit mode option
- Button combinaton: **Start + R + Z + …**
  - **… A + B** : trigger a reset
  - **… C-ri** : de-blur for 240p on
  - **… C-le** : de-blur for 240p off
  - **… C-up** : 15bit mode off
  - **… C-dw** : 15bit mode on

Notes:

- Controller is only read if the actual game reads the controller
- Switching de-blur overwrites the heuristic estimation.
  De-blur option is reset with each triggered reset and power cycle
- 15bit mode is reset with each power cycle
- N64A: buttons has to be enabled in menu

Menu (N64A only)



Open menu button comb.

- To open the menu press **Ri + L + R + C-ri**
- Use **Le + L + R + C-le** at every menu page to close menu or simply press **B** until menu has been closed
- Once the menu is open you have the following sub menu pages available:
  - Video-Info (shows current video processing state)
  - VI configuration (configure video interface (2pages))
  - Miscellaneous config (configure in-game routines, filter addon and show experimental testpattern)
  - Load/Save (save config to flash and load from flash, jumpers and N64 defaults)
  - About… (some information about the modding kit)
  - Acknowledgment... (thankfully words to some people who helped me)
  - License... (license)

Menu (N64A only) – Navigation

To navigate through the menu you are can use …

- **D-pad up** or **down** to *select the submenu* you want to enter / *option* you want to modify
- **D-pad left** or **right** to *change an option*
- **A** or **D-pad Ri** to *enter a submenu*
- **B** to *leave a submenu* (close menu on top level menu)
- **Le + L + R + C-le** to *close menu*
- **Z + L** to *mute the menu* (menu overlay disappears as long as combination is pressed)

- Instead of the **D-Pad** you can also you the **analog stick**.
- **Options are shown in yellow if they do not match the setting stored in flash memory!**

Menu (N64A only) – VI config. 1

Menu Point	Description
NTSC/PAL awareness	Decide whether you want to configure linemultiplier options for both NTSC and PAL games or whether you want to apply different settings (If you opt for global settings, the NTSC setup is applied and showed in menu)
Linemultiplier	Change between NTSC and PAL submenu (does not applied if NTSC/PAL awareness is off)
- 240p/288p settings	Enters linemultiplier submenu for progressive video input
- 480i/576i setting	Enters linemultiplier submenu for interlaced video input
V/H position/timing	Enters positioning and timing submenu (Options only available for enabled linemultiplication)
Gamma Value	Apply gamma curve on output with $I = (I_{\text{in}})^{\gamma}$, where I_{in} is the input intensity and γ the gamma value (In other modifications, this is called 'gamma boost' with the difference that the value is inverted)
VI config page 2	Enters VI configuration page #2

Menu (N64A only) – VI config. 2

Menu Point	Description
Color Space	Selects between RGBS (RGB + separate Sync), RGBS/RGsB (RGB with sync on green) and YPbPr (Component video) (Separate sync stays enabled for all modes)
Exchange R&B out	Changes outputs of R and B (Pr and Pb) (Option was introduced just for a few experimental flex cables where I exchanged red and blue)
LowRes.-DeBlur	Select between On and Off (low resolution deblur does not apply for interlaced content) - On activates VI-DeBlur, which may improve image quality for 320x240 (low resolution) content but decreases image quality for 640x240 (high resolution) content - Off deactivates VI-DeBlur
- power cycle default	VI-DeBlur default setting after a power cycle
15bit mode	Select between On and Off - In 15bit mode the color depth is reduced from standard 21bit (7bit per color) to 15bit (5bit per color) b simply cutting off the LSBs
- power cycle default	15bit mode default setting after a power cycle
VI config page 1	Enters VI configuration page #1

Menu (N64A only) – Config (240p/288p)

Menu Point	Description
Enable Linemultiplier	Disables or Enable Linemultiplier by choosing between - LineX off: disable linemultiplier - LineX2: enable linedoubler (every scanline is outputted twice) - LineX3: enable linetrippler (every scanline is outputted three times | **needs VPLL to be enabled** | **not available in PAL**)
- Video PLL	Submenu to test and activate/deactivate the video PLL. The clock source of the N64 is rather weak in order to drive long installation wires. Depending on you installation the PLL does not provide a stable lock, which means that you cannot use LineX3 then.
Use Scanlines	Enables scanlines for linedoubled and linetrippled 240p content
- Method	Choose between simple and advanced - simple: Scanline calculated based on previous (odd scanline ID) or next (even scanline ID) drawn line - advanced: Scanline calculated based on both, previous and next, drawn line
- Scanline ID	Choose between even (scanline before actual drawn line) and odd (scanline after drawn line)
- Scanline Stength	Select the scanline strength with $I = s \times I_{\text{in}}$, where s is the scanline strength and I_{in} the pixel intensity
- Hybrid Depth	Make scanline strength pixel-intensity dependent - 0% means that scanlines are calculated as setup - 100% means that scanlines strength is reduced to 0 for maximum pixel intensity - (> 100%) means that scanline strength is reduced to 0 prior to maximum pixel intensity - (< 100%) means that scanlines are always drawn but slightly reduced with increasing pixel intensity

Menu (N64A only) – Config (480i/576i)

Menu Point	Description
De-Interlacing (Bob)	Disables or Enable Linedoubler by choosing between - off: disable linemultiplier - on: enable linedoubler (every scanline is outputted twice)
- Field-Shift Fix	Shifts all fileds half of a pixel up and down between linedoubled even and odd field reducing some typicall up-down-flickering of simple bob deinterlacing in most cases
Use Scanlines	Enables scanlines for linedoubled / bob-deinterlaced 480i content
- Method	Choose between simple and advanced - simple: Scanline calculated based on previous (odd scanline ID) or next (even scanline ID) drawn line - advanced: Scanline calculated based on both, previous and next, drawn line
- Scanline ID	Choose between even (scanline before actual drawn line) and odd (scanline after drawn line)
- Scanline Stength	Select the scanline strength with $I = s \times I_{\mathrm{in}}$, where s is the scanline strength and I_{in} the pixel intensity
- Hybrid Depth	Make scanline strength pixel-intensity dependent - 0% means that scanlines are calculated as setup - 100% means that scanlines strength is reduced to 0 for maximum pixel intensity - (> 100%) means that scanline strength is reduced to 0 prior to maximum pixel intensity - (< 100%) means that scanlines are always drawn but slightly reduced with increasing pixel intensity

Menu (N64A only) – Config (Position)

Menu Point	Description
Settings for:	Selects configuration parameters between differnt modes (mode depends on NTSC/PAL, progressive/interlaced and LineX)
Vertical shift	Shifts image up (positive values) and down (negative values)
Horizontal shift	Shifts image left (negative values) and right (positive values)
DeJitter for PAL LX2	Removes line width variation in non-visible vsync area as good as possible (not by 100%, half of a pixel variation remains as this feature does not change the console timing at all) Experimental feature, which may not work for all games -> please report incompatibilities on GitHub!
Reset values	Resets values to defauls (all-zero) Reset is only available if “settings for:” does not select the current mode

Menu (N64A only) – Misc. Config

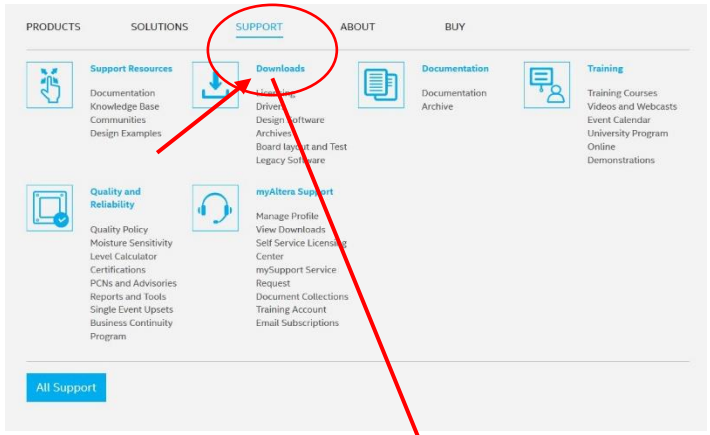
Menu Point	Description
In-Game Routines	
- Reset	Enables in game reset button combination (St. + Z + R + A + B)
- Quick Access	Enables in-game routine for vi-deblur and/or 15bit mode toggle - VI-deblur: St. + Z + R + C-ri (option on) and St. + Z + R + C-le (option off) - 15bit mode: St. + Z + R + C-dw (option on) and St. + Z + R + C-up (option off)
Filter Add-On	Select the filter cut-off for the filter add-on board - Jumper J1.1 has to be closed to have access to next option
- Filter Cut-Off	Select between Auto, 9.5MHz, 18MHz, 36MHz and bypassed
Show Test-Pattern	Show an experimental test pattern (checkered board atm) in 240p or 480i mode (depending on actual input)

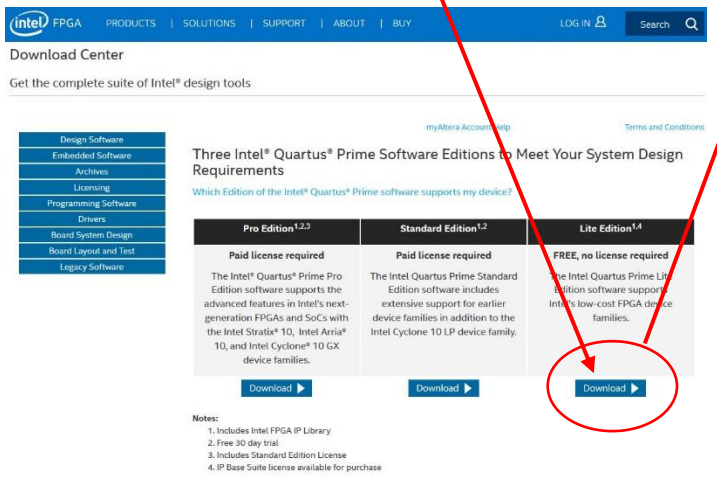
Menu (N64A only) – Load/Save

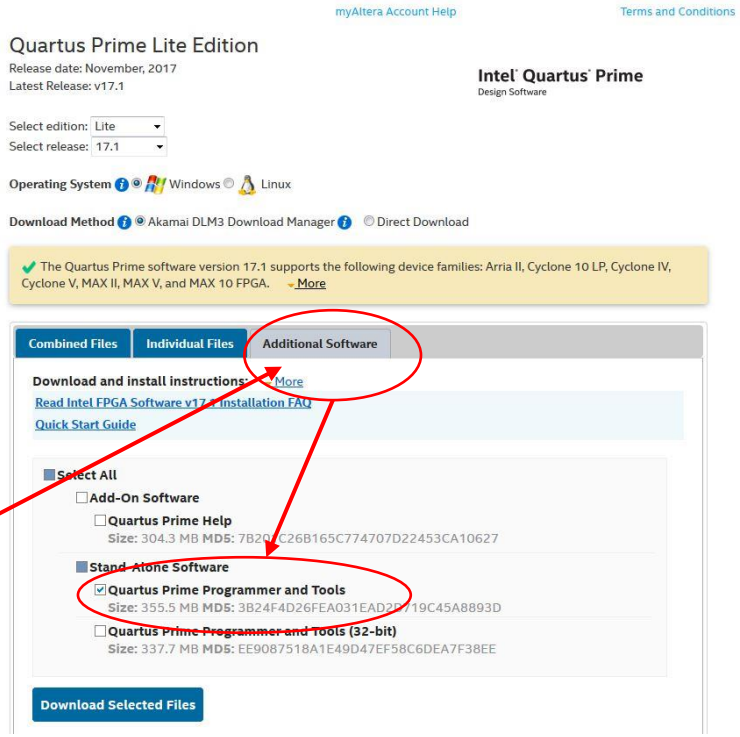
Menu Point	Description
Save: **- Configuration**	Save actual configuration to flash memory. This is needed to have actual settings applied after a power cycle.
Load	Different options to load settings
- Last Configuration	Load the setting which are stored in flash memory
- Defaults from Jumper Set	Load defaults which are determined by PCB jumper settings
- N64 Standard	Load N64 standard setting, e.g. turn off linemultiplier These settings are also loaded if you hold down reset button while power cycle the system.

Firmwareupdate

-- Download the software tools --







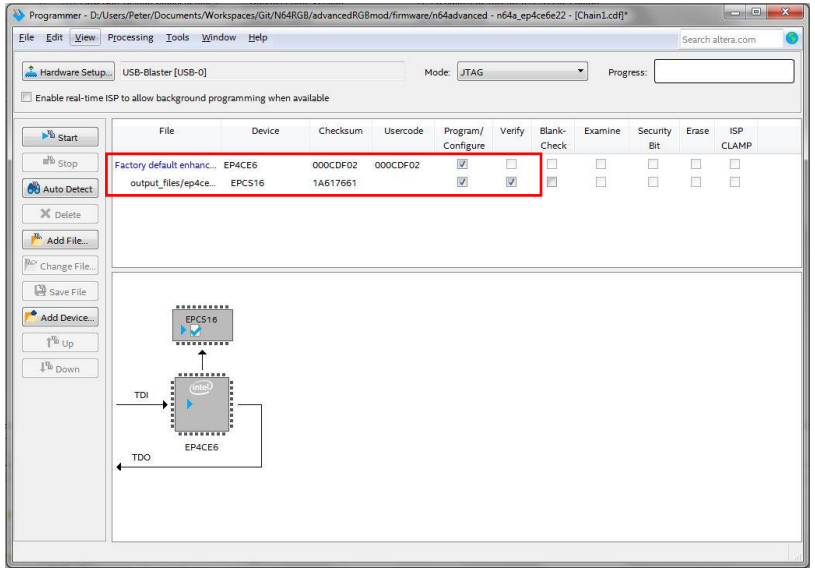
1. Go to altera.com and select “Downloads” from the “SUPPORT” menu
2. Select “Download” from the “Light Edition” column of the “Design Software” (preselected)
3. Go to “Additional Software”
4. Download the “Quartus Prime Programmer and Tools software

If you have the Quartus Prime edition installed, the programmer tools are already available on your computer.

Date: 22.11.2017
Website layout may change

Firmwareupdate

-- Update your N64 Advanced --

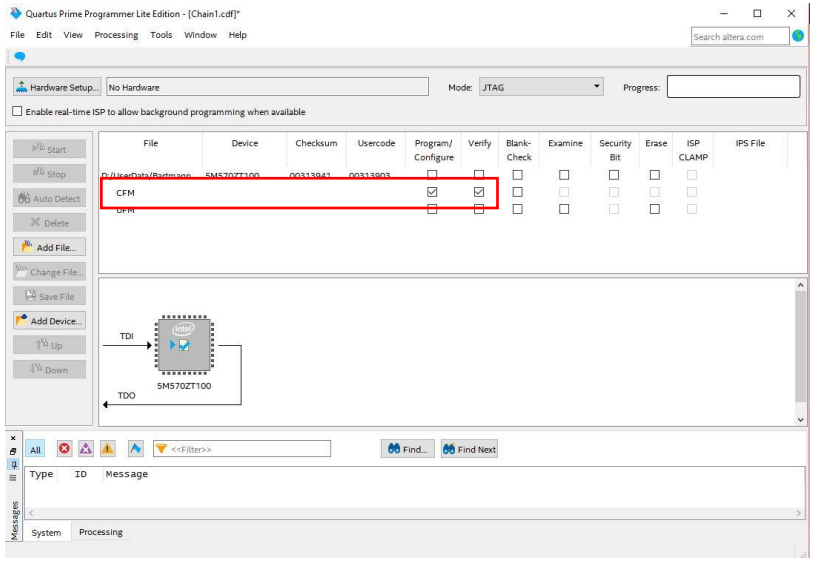


- Connect your PC with the USB Blaster and the USB Blaster with the modding kit J5
- Download the actual firmware build from the GitHub repository folder advancedRGBmod/firmware/output_files/*yourFPGA*/n64a_*yourfpga*.jic
- Open the programmer tool
- Load the previously downlaoded JIC file with “Add File…”
- Check “Program/Configure” for the FPGA and “Program/Configure” and “Verify” for the Flashdevice
- Make sure that the USB Blaster is selected in the “Hardware Setup”
- Click on “Start” (the N64 has to be powered for the reference voltage) and wait for the process to be finished
- Power cycle your N64 and enjoy :)

Quartus Prime Lite v17.1

Firmwareupdate

-- Update your N64 RGB --



- Connect your PC with the USB Blaster and the USB Blaster with the modding kit J10
- Download the actual firmware build from the GitHub repository folder generalRGBmod/firmware/output_files/v#/n64rgb_*yourcpld*.pof
- Open the programmer tool
- Load the previously downlaoded POF file with “Add File…”
- Check “Program/Configure” and “Verify” for the CFM part of your device
- Make sure that the USB Blaster is selected in the “Hardware Setup”
- Click on “Start” (the N64 has to be powered for the reference voltage) and wait for the process to be finished

Quartus Prime Lite v17.1