--- PSRAM Architecture ---
Pico1, Pico2 and all PSRAMs are connected to a shared QSPI bus.
Pico1 and Pico2 are connected to a 3-bit, 8 output demux.
Demux is connected to each PSRAM's CS pin.
Pico2 writes to PSRAM.
Pico1 reads from PSRAM.
Pico2 can hold Pico1 in reset so there is no bus arbitration needed.

--- N64 ROM ---
Pico2 holds Pico1 in reset, fills PSRAM with data.
Pico2 reads data from SDCard via WiFi module (running custom firmware).
Pico1 handles N64 bus requests, reads data directly from PSRAM.

--- N64 SRAM ---
Pico1 handles bus request, read/store in internal SRAM.
Stores to external flash and/or SDCard via Pico2.

--- Pico1 -> Pico2 communication ---
Pico1 asserts Pico2_CS.
Pico2 is interrupted and handles the request as an SPI peripheral.
Cart/Pico1 can request data from WiFi and/or SDCard via Pico2.

--- Firmware update ---
1. User connects USB to Pico2
Pico2 firmware is updated with UF2. Easy solution: use boot select button (ROM BL).
Pico1 firmware is updated from Pico2 (Pico1 FW is bundled inside) via Pico1_FLASH pins (easier than SWD)
WiFi Module is updated somehow with ESP32 serial downloader? TODO: Check how this works

2. User has a special file on SDCard
Pico2 scans SDCard during boot, finds file, updates itself (SCARY!)
  - Maybe wifi module should connect to Pico2_FLASH + Pico2_RST instead?

--- WiFi module role ---
Pico2 can initiate WiFi stuff
WiFi module can load ROMs from network to SDCard
WiFi module can save SRAM saves to network

---------- Open questions

USB Debugging? Doable or does it mess up irq?
Both Picos need to be very responsive.

WiFi access while game is running:
Problematic, since accesses may happen at any time.
Maybe possible between ALEH/ALEL signals go high after READ

Debug/Print via WiFi?

Real debugging using INT?

POWER BUDGET

2x RP2040	2*100(?)	= 200mA (let's use max rating @133MHz)
8x PSRAM ly68l6400slit	1 * 40	= 40mA (only draws I during read/write)
	8 * 1	= 8mA (idle current)
1x ESP32	345 mA (!) during wifi tx	
1x SDCard (SPI mode)	150mA	

Total: 200+40+8+345+150 = 743 mA

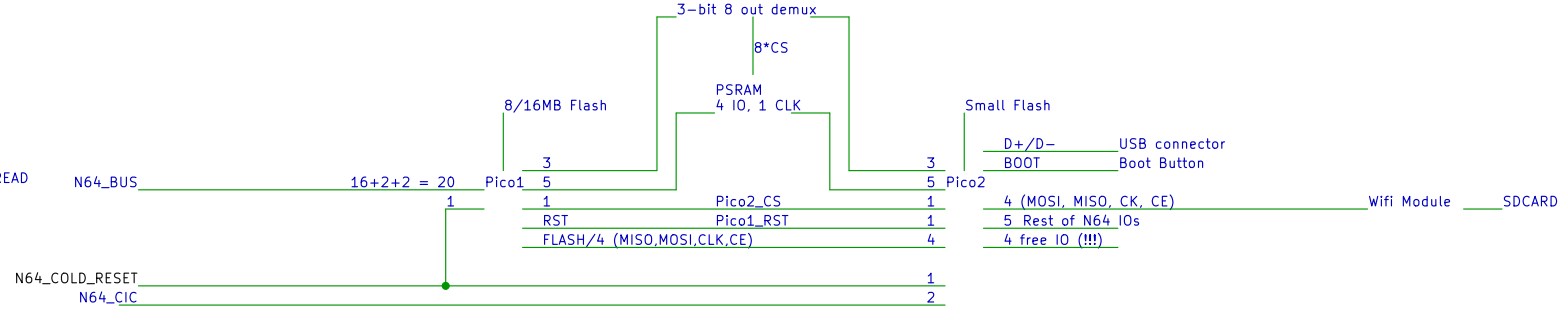
Need _at least_ 750mA supply, 1A recommended.

BOM: @ qty 1 (will be cheaper when producing 100x +)

2x RP2040	= $2	
WiFi Module (ESP32)	= $3	
MC74AC138	= $1	
SDCard slot	= $0.5	
4 x LY68L6400SLIT	= $11	
USB conn.	= $0.5	
SPI Flash (16MB)	= $1.1	W25Q128JVSIQ
SPI Flash (2MB)	= $0.5	W25Q16JVSSIQ
2 x Crystal 12MHz	= $0.5	XYDBPCNANF-12MHZ / C521567
5V -> 3.3V 1A LDO	= $0.34	SSP1117-3.3V / C277892
TBD 5V/3.3V switch	= $1	BSS84 is too weak
PCB	= $1	

Total 2+3+1+0.5+11+0.5+1+1+0.5+0.5+3.34+1+1 = $26.34

Customer price = 3 * BOM = $79.02



Pico1 Pin Count = 20+1+1+5+3 = 30

Pico2 Pin Count = 3+5+1+1+4+1+2+5+4 = 26

Pico SRAM = 264 kB
Let's use 128kB dedicated
for save game SRAM

PSRAM = 4 I/O + CLK
CE = 1 per chip

CE demux 3 bit -> 8 CE

MC74AC138 fast 3-bit 8 output demultiplexer (active LOW)

Alternative demux
sn74lvc138a-q1

Sheet: /Concept2/
File: Concept2.kicad_sch

Title:

Size: A4 | Date: | **Rev:**
KiCad E.D.A. kicad (6.0.6) | Id: 3/4