GREENWAVES TECHNOLOGIES

Application Note
AN003

GAP8 HARDWARE INTEGRATION GUIDE

Version : Rel.1.1
Date : March, 2020
Author : Xavier Cauchy

CONTENTS

DISCLAIMER

This information is subject to change without notice.

Information on this document is provided “as is” without any warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, suitability for a particular purpose, or non-infringement. The information provided in this document is intended for informational purposes only. Information may be changed or updated without notice.

1. INTRODUCTION

This application is intended to help hardware designers integrate GreenWaves GAP8 ultra-low IoT Application Processor chip into their board design.

Note: Marks noted *« [Ref.X]* » in this text point to references provided at the end of the document.

2. REFERENCE CORE DESIGN

Figure 1 provides an example PCB design around GAP8, with recommended connections to external memory, to crystal, to passives for internal DC-DC converter and to power supply sources. Sheet 1.c shows two variants that differ in the type of external memory used : HyperBus memory or Quad-SPI memory – refer to section 6 for futher information on memory options.
This is a generic example that needs to be tuned to your specific system. It assumes LVDS interface is not required.
The next sections will offer advice and recommendations on specific aspects of GAP8 hardware integration.

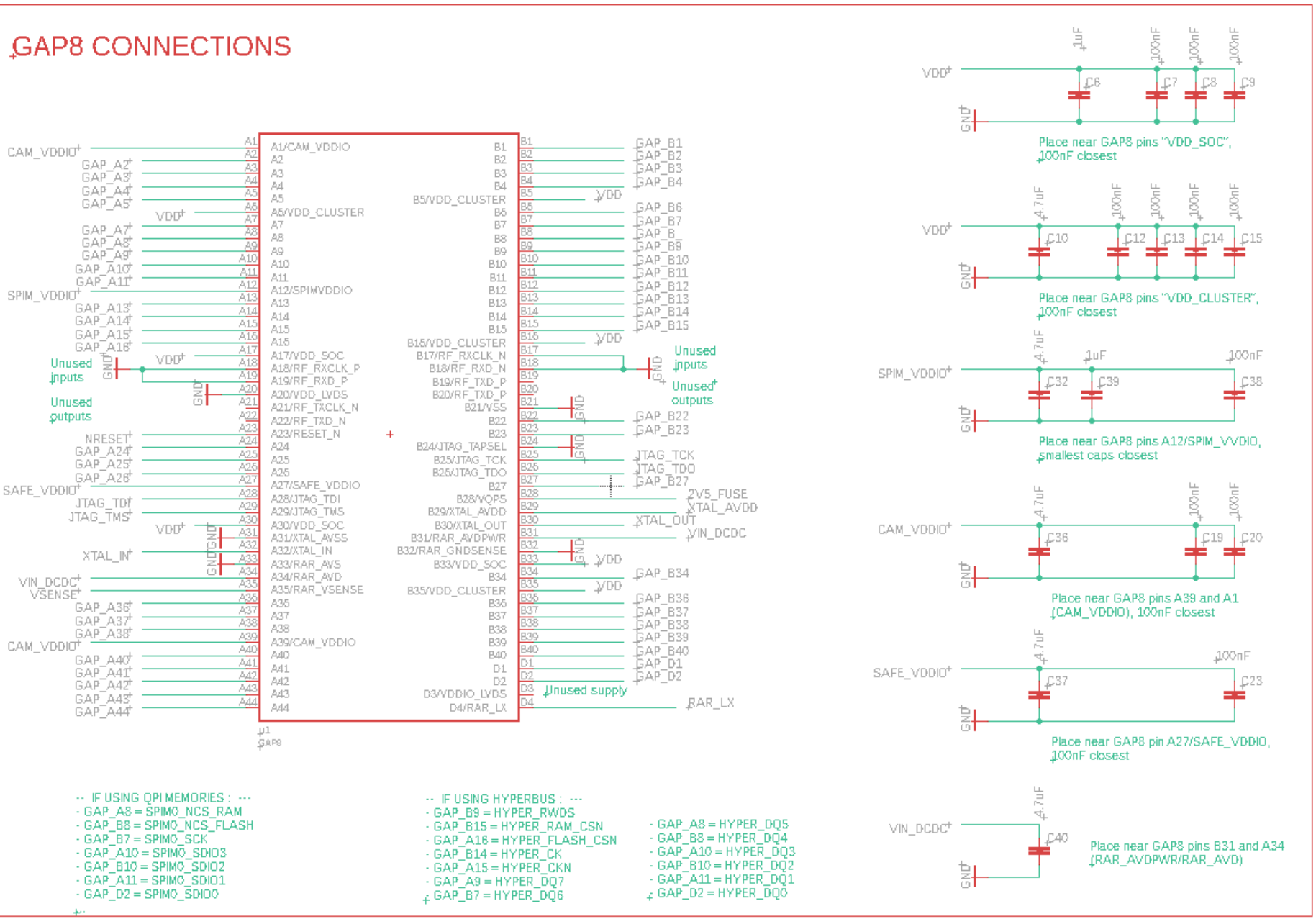


Fig. 1.a - Generic Reference Design – Part 1/4

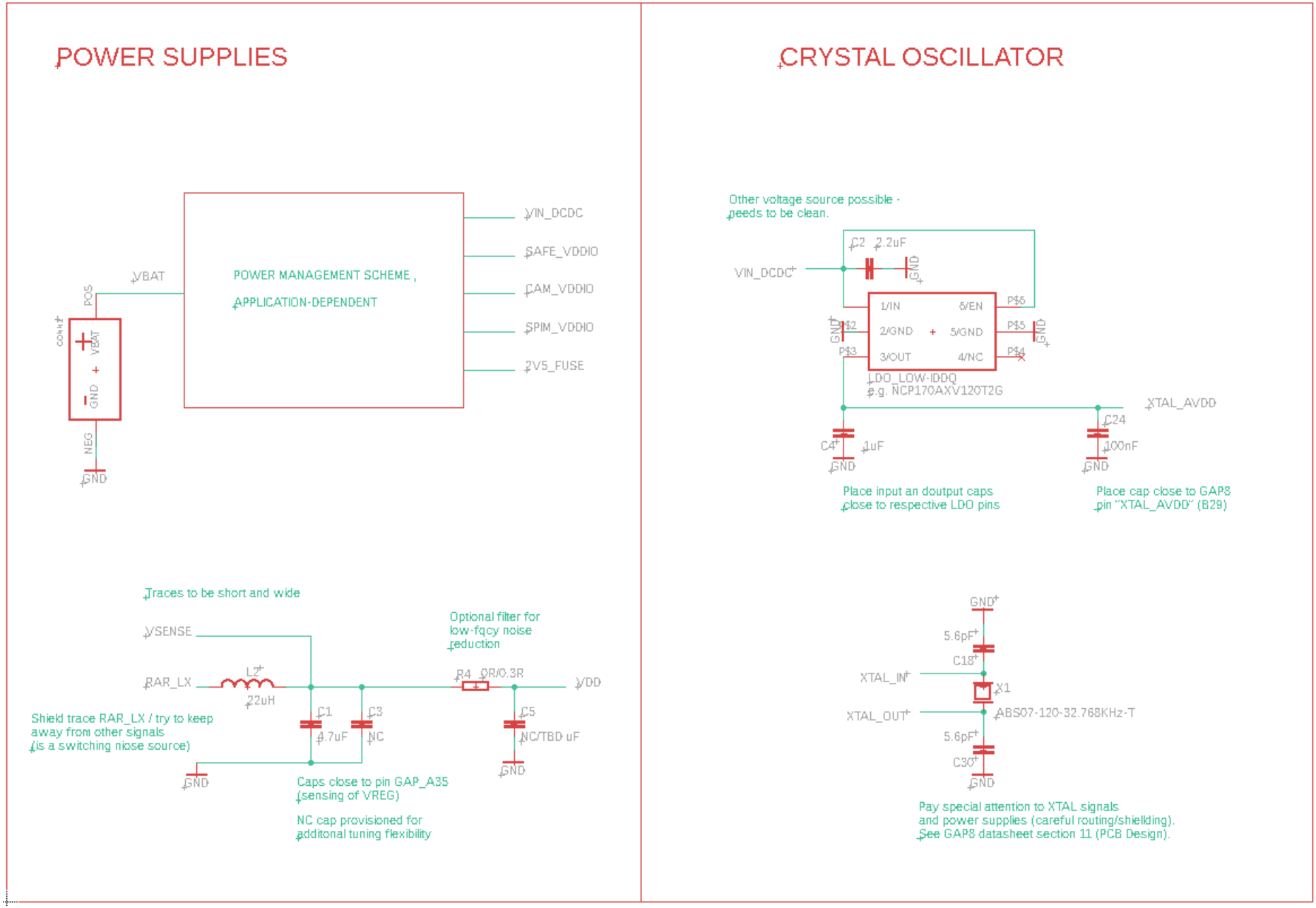


Fig. 1.b - Generic Reference Design – Part 2/4

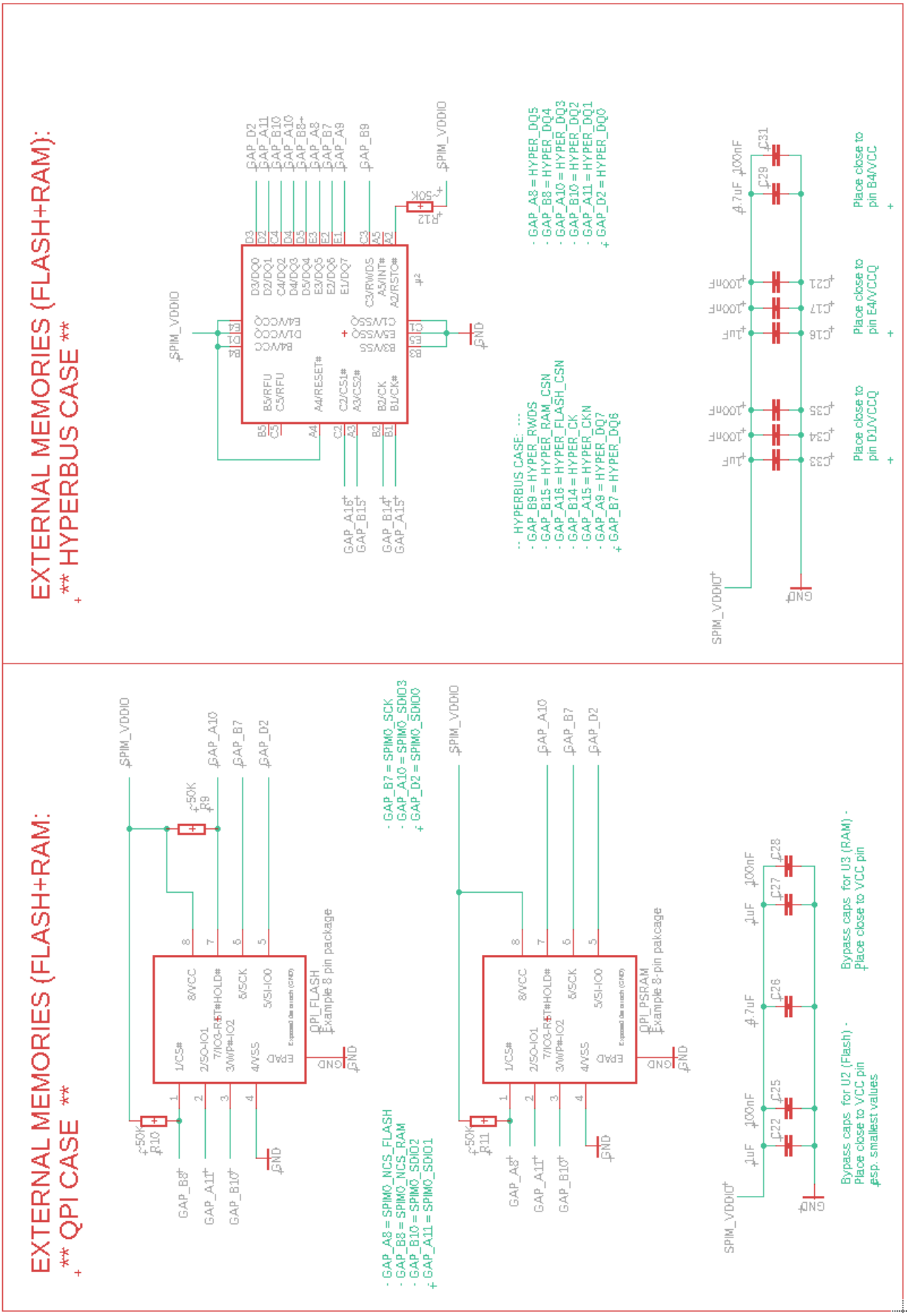


Fig. 1.c - Generic Reference Design – Part 3/4

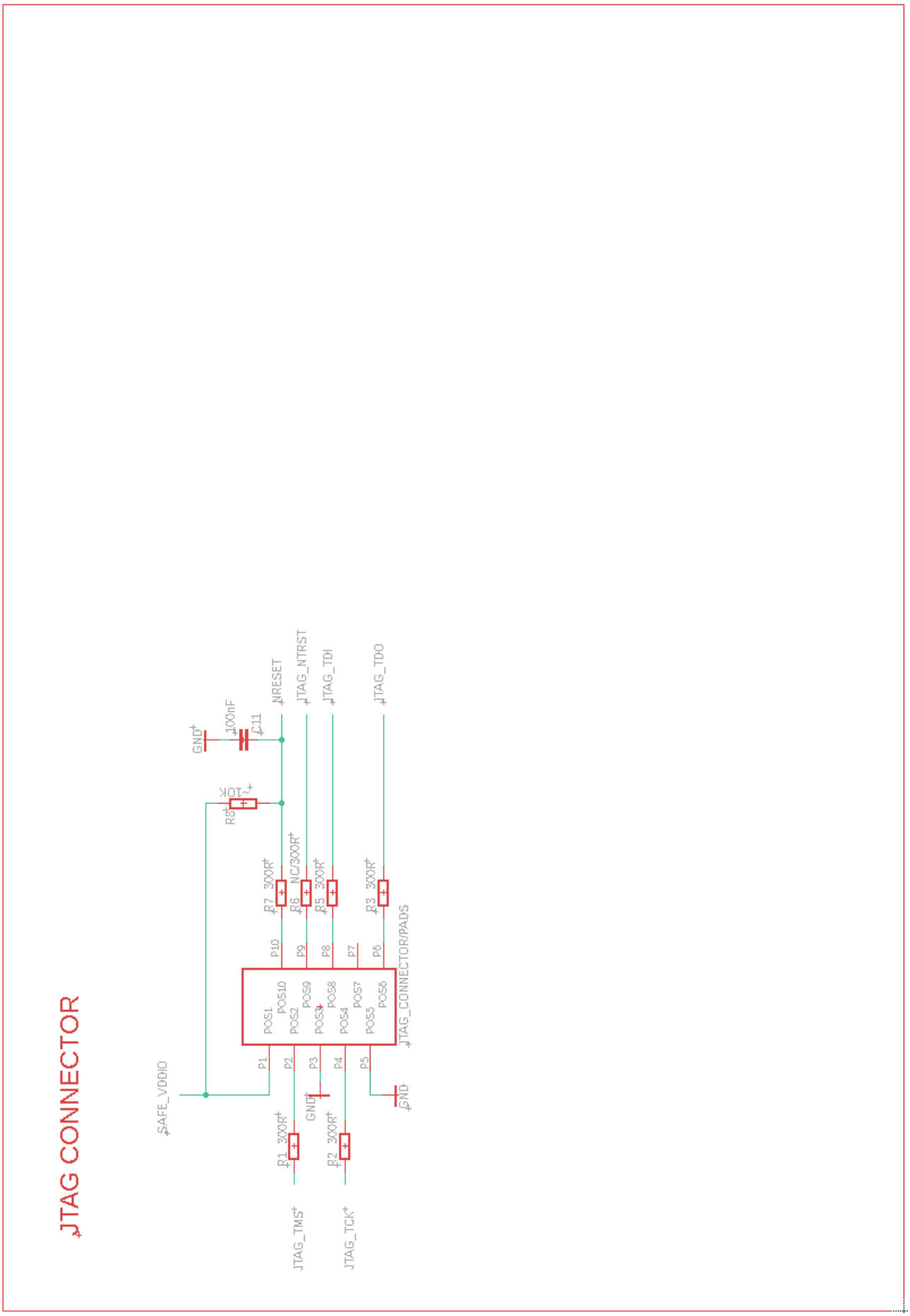


Fig. 1.d - Generic Reference Design – Part 4/4

3. POWER MANAGEMENT

> Core power supply

GAP8 embeds a **DC-DC converter** that takes an **input voltage (*Vin* in Figure 1) between 1.6V and 3.6V**, to be provided on pins **RAR_AVD (A34)** and **RAR_AVDPWR (B31)** – using the naming and pin numbering provided in section 4 of GAP8's datasheet *[Ref.1, « Pinout and Pin Description »]*.
This input voltage would typically come from the system's battery, either directly or through some voltage conversion stage.
The internal DC-DC converter generates a switching voltage on pin **RAR_LX (D4)**. As illustrated in Figure 1 and GAP8's datasheet *[Ref.1]*, this signal must be passed through an LC tank to obtain a stable voltage (*VREG*) for GAP8's internal logic (FC and Cluster). This output voltage can be set between 1V and 1.2V (in steps of 50mV) under software control. This output voltage must be fed back to the DC-DC through pin **RAR_VSENSE (A35)** and also used as GAP8 core power supply, injected though pins **VDD (A17, A30, B33)** and **VDD_EXT_CLUSTER (A6, B5, B16, B35)** (possibly through some noise filtering stage).
The switching signal on RAR_LX is a potential noise source (agressor) to surrounding signals ; it is therefore recommended to take adequate precautions. In addition, traces between GAP8 and the inductance and between GAP8 should be kept as short as possible. A specific section on DC-DC follows further down this document.

> Crystal oscillator Power Supply-

A crystal oscillator running off a 32.768KHz crystal generates a master clock from which all internal and peripheral clocks are then derived. It is powered through a dedicated pair of supplies : **XTAL_AVDD (B29)** and **XTAL_AVSS (A31).** XTAL_AVDD is **1.2V nominal** but may go down to 0.9V.

XTAL_AVDD would typically be generated by a dedicated LDO performing a down-conversion from VBAT (primary system power supply, typically from battery). In this case the oscillator is always running and an ultra-low quiescent current LDO is beneficial if sleep mode current is of importance.

In some cases, it is possible to use an alternative, lower cost solution, which consists of supplying XTAL_AVDD with VREG, the regulated output from internal DC-DC converter. However this comes with some restrictions to keep in mind :
-this should preferably be limited to scenarios where VREG will stay between 1V and 1.2V, i.e. GAP8 is always on and never programmed into sleep mode (in which case VREG would drop to 0.8V).
- VREG will not be as clean as VBAT+LDO, which may impact the quality (esp. jitter) of the reference clock produced by the 32KHz oscillator.
- finally, in a scenario where system power (including Vin) would be completely switched off in system sleep mode, using VREG to power the crystal oscillator would mean the oscillator has to restart at each system wake-up, which typically takes several hundreds of ms for a 32KHz crystal.

> I/O ring -

The I/O ring is split into 4 sections as shown in Figure 2, each powered from a dedicated supply : SAFE_VDDIO, SPIM_VDDIO, CAM_VDDIO and VDDIO_LVDS.

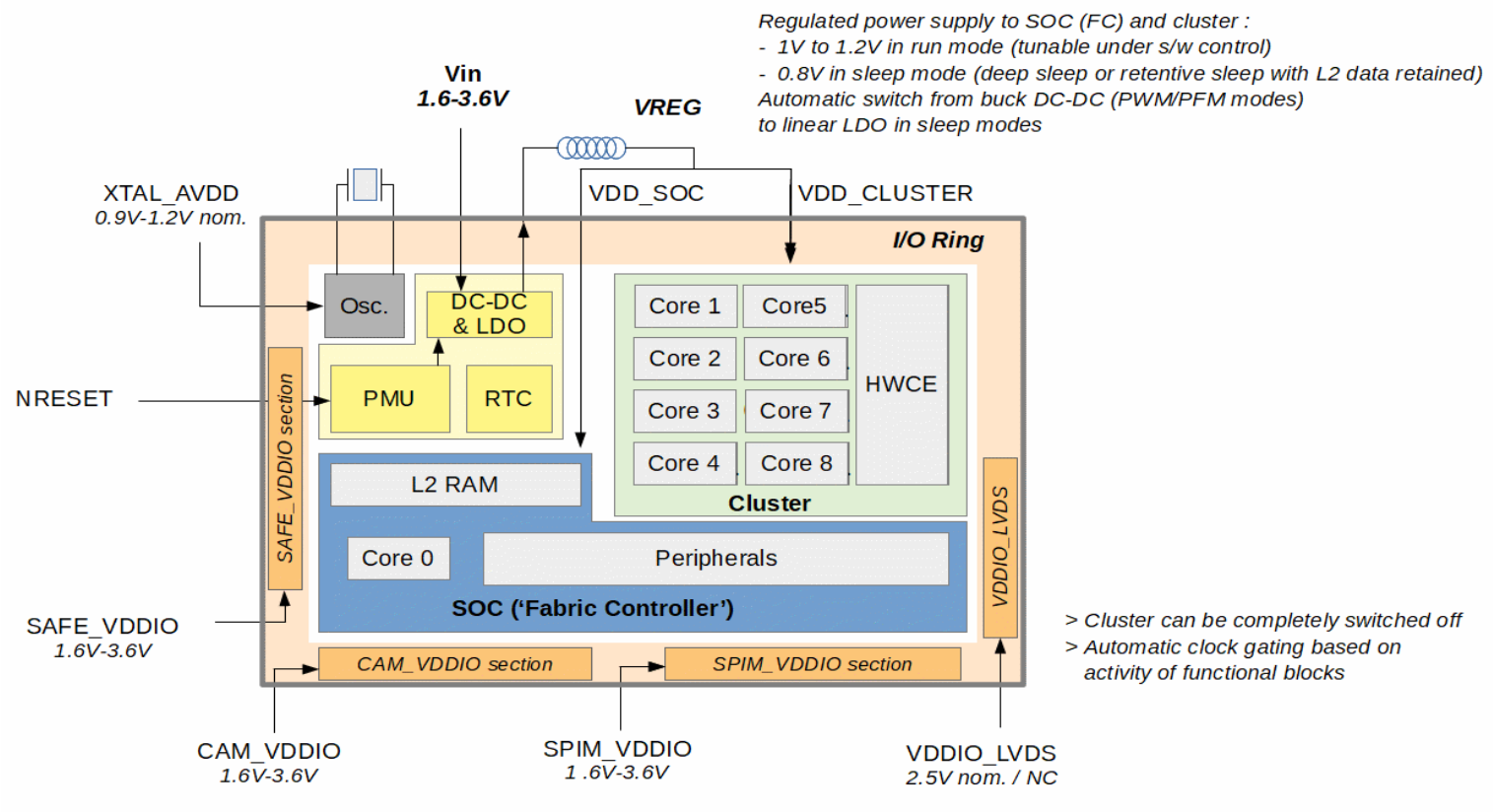


Fig. 2 : Power Domains in GAP8

- **VDDIO_LVDS (D3)** and **VDD_LVDS_1P2V (A20)**:
If the LVDS interface is required in the application, VDDIO_LVDS needs to be fed with a **2.5V**+-10 % power supply. Otherwise, it is best to leave VDDIO_LVDS **unconnected** (as the PHY of this interface consumes static power when powered). In addition, in the former case, pin VDD_LVDS_1P2V (A20) should be supplied with 1.2V+-10 %; in the latter case (LVDS interface not required) it can be grounded.

- **SAFE_VDDIO (A27), CAM_VDDIO (A1 & A39)** and **SPIM_VDDIO (A12)**:
the other I/O ring power supplies, listed above, can be freely chosen **between 1.6V and 3.6V**, depending on application requirements (in particular, according to I/O voltage levels of memories and sensors interfacing with GAP8).

GAP8 I/Os can sustain up to 3.6V without damage. However from a functional point of view it must be kept in mind that applying a voltage greater than the supply voltage of the I/O (SAFE_VDDIO, CAM_VDDIO or SPIM_VDDIO) will cause the ESD protection diode to become conductive (Fig.3). Therefore, a series current limiting resistor should be provisioned if it is anticipated that voltage on I/O pin may exceed the supply voltage of that particular IO.

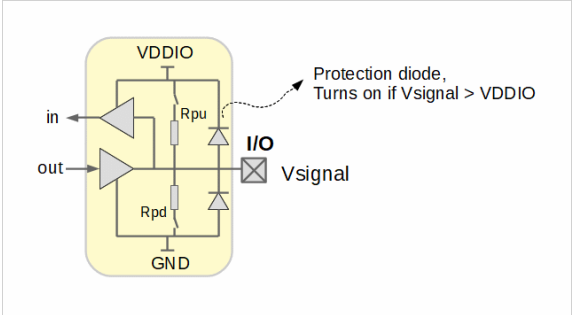


Fig. 3: Simplified View of I/O structure

An additional power supply, VQPS, is dedicated to programming or e-fuses, see below.

> Pin VQPS (B28) -

This is the programming voltage for the embedded eFuse cells. It needs to be present when programming (blowing) the eFuse to permanently set eFuse bits to either 0 or 1 – and it is not required outside fuse programming. The required voltage is **2.5V+-10 %**. Current draw during programming is **32mA** nominal (35mA worst case).
eFuse programming enables to adjust different settings and behaviors of GAP8. In particular, getting GAP8 to boot from external Flash on power-up rather than from JTAG is achieved through the programming of a specific eFuse bit.
This means **it is mandatory to be able to program the eFuse** and therefore have some means to provide VQPS=2.5V+-10 % **at least on production boards** – and in fact on any board that needs to be usable without a JTAG probe and host PC attached to it on power-up.

> Bypass capacitors -

All power supply pins should be properly bypassed to GND through low ESR (Equivalent Series Resistance) caps, placed as close as possible to the pins they decouple.
Ceramic capacitors with X5R or X7R dielectric type are recommended. Keep in mind this type of capacitor exhibits dependency to DC bias, meaning the actual capacitance matches nominal capacitance at 0V bias but (especially in small form factors such as 0402) can drop rather sharply as the DC bias applied to the cap increases.

> Power Supply Sequencing -

Fig. 4 describes the required sequencing when starting up power supplies.

Essentially :
- SAFE_VDDIO may rise up to 150ms after *Vin_dcdc* i.e. RAR_AVD/RAR_AVDPWR (GAP8 pins A34 & B31) rise,
- NRESET must be up no later than 150ms after *Vin_dcdc (*RAR_AVD/RAR_AVDPWR) rises,
- CAM_VDDIO may be switched on and off at any time,
- the timing allowed on SPIM_VDDIO depends on fuse settings. Those control the latency before the primary boot code (executed from internal ROM) jumps to secondary boot code fetched from external Flash – at which point the Flash interface, which is on SPIM_VDDIO, must obviously be up.

The strictest requirement is t_4= 500us maximum from NRESET rising to SPIM_VDDIO stable. It can be extended up to 1 second with adequate fuse settings. Note however than adding latency before first access to Flash means reboot from deep sleep modes (no code kept alive in internal RAM) will also incur additional delay.

If SAFE_VDDIO and NRESET are up before Vin_dcdc, then t_4 should be taken as the time between *Vin_dcdc* up and SPIM_VDDIO up

XTAL_AVDD and I/O power supplies (SAFE_VDDIO, SPIM_VDDIO, CAM_VDDIO) may be up before *Vin* (RAR_AVD) rises.

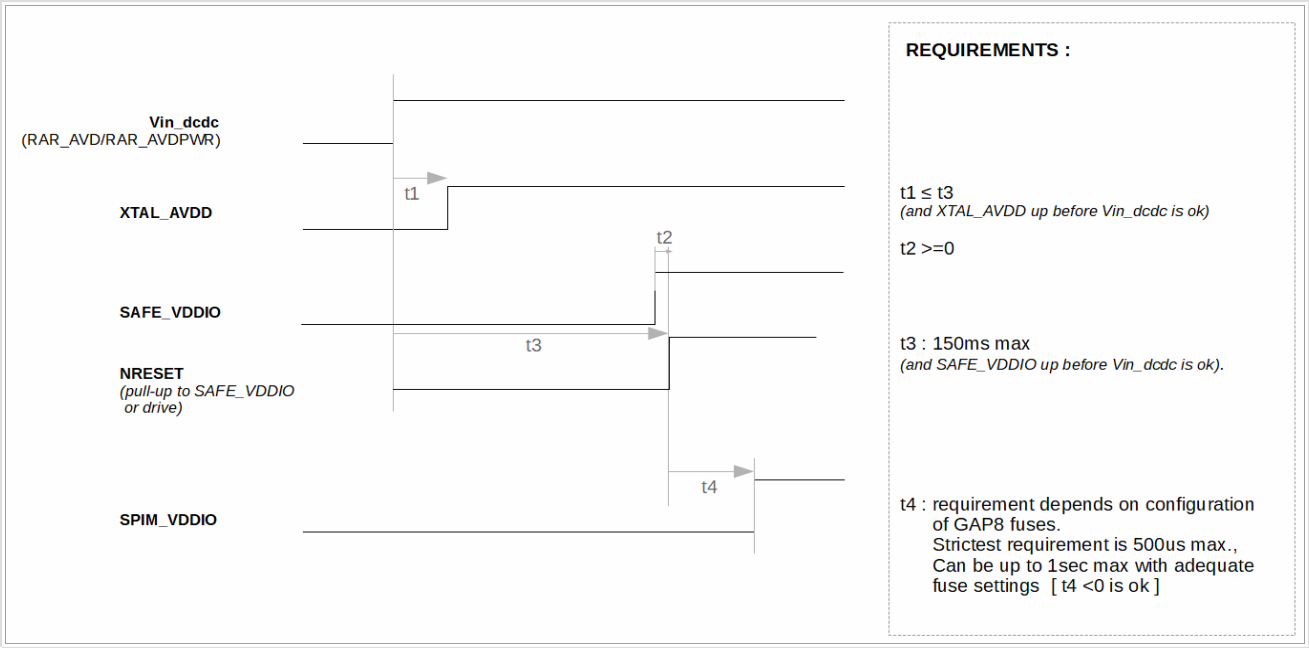


Fig. 4 : Power Supply Sequencing Diagram

4. CRYSTAL OSCILLATOR

GAP8 embeds an oscillator tuned to operate with a 32.768KHz crystal connected across pins **XTAL_IN (A32)** and **XTAL_OUT(B30)**. It is advised to select a crystal with **low ESR** (<60Kohm) and **low load capacitance**, and to place the crystal close to GAP8, to limit parasitics as much as possible.
A suitable choice is for instance Abracon's ABS07-120-32.768KHZ-T, loaded with 5.6pF on each pin (ceramic capacitors of C0G dielectric type are a suitable choice).

Oscillator start-up time is of ther order of 400ms. Therefore for fast system wake-up, it is recommended to keep the crystal oscillator powered (through XTAL_AVDD, pin B29) when the system is in deep sleep – so that overall reboot time does not incur time for crystal to reconverge.

> PCB Placement and Layout Recommendation for Xtal oscillator :
Refer to section 11.1 of GAP8's datasheet which provides PCB design guidelines related to the crystal and the oscillator. In particular, it is recommended to implement a guard ring around the crystal + load capacitors and dedicate a ground « island » to this area.

5. INTERNAL DC-DC CONVERTER

The internal DC-DC requires a number of external passive components to operate properly, as visible in the reference schematics (Fig.1) and explained in section 11.2 of GAP8's datasheet.
A **22uH** inductor value is recommended. It is advised to select a shielded, **low DCR** inductor (1 ohm or better is a reasonable value) – however lower equivalent DC resistance (DCR) typically comes at the expense of larger overall dimensions so this should be weighted against application needs and constraints. Low DCR helps achieve good conversion efficiency ; shielding limits the amount of noise radiated by the inductor and therefore risks to pollute neighboring signals.

> Sensitivity to input level at start-up - Precaution :
For proper start-up of the internal DC-DC, it is important that voltage at the input of the DC-DC converter is 0V before power-up voltage is applied. Otherwise the DC-DC may not converge properly and may fail to deliver 1.2V at its ouput.

Therefore, if the power supply at DC-DC input can be switched off and on, the implementation should ensure the voltage at the ouput of the switch gets quickly back to 0V when turned off, taking into account any capacitive effect. This may for example call for usage of a switch with output discharge capability when turned off.

> PCB Placement and Layout Recommendation for internal DC-DC converter :
Section 11.2 of GAP8's datasheet provides some PCB design guidelines related to the internal DC-DC regulator (« RAR »).
In addition, it is worth keeping in mind that pin D4/RAR_LX is the unfiltered switching output of the internal DC-DC converter and the associated net will therefore be a potential noise source – so it is important to keep it away from sensitive signals (and perhaps shield it using some grounded guard ring).

6. EXTERNAL MEMORIES

6.1 - Choosing the right type of memory

GAP8 embeds a 512KByte L2 volatile memory (RAM) that can be used to store data and code. At boot, application code would typically be obtained from an external Flash. An external RAM may also be required to store amounts of data larger than the L2 can hold, depending on algorithm requirements.

The required capacity of the external Flash depends on code size and on the size of other data that may need to reside in non-volatile memory (for example, set of coefficients to be used by the algorithms embedded in the user code).

Usage of an external RAM is optional and directly depends on the requirements of the applicative software, as does sizing of the RAM. Some algorithms might be fairly lightweight in that respect and can live with the 512KB internal RAM, others will be more demanding and will require extra RAM.

> HyperBus vs. Quad-SPI Memories :

GAP8 is able to interface with external memory chips either through a HyperBus interface or through a Quad-SPI interface (which is a serial SPI interface augmented with the ability to transfer 4 parallel bits at a time, thereby significantly improving the available bandwdith vs. a single SPI interface running at same serial clock speed). HyperBus and QPI share a common set of pins (SPIM0 interface of GAP8). The HyperBus specification is available on Cypress's web site *[Ref.3]*.

Here are some facts to help decide which is best suited for your own application :

- The HyperBus interface can run at up to 125MHz in DDR mode with 8 data bits per cycle. The Hyperbus clock rate is always half the FC (SOC) core clock speed – so, for example, running the memory at 100MHz requires to run the FC at 200MHz – and, conversely, running the FC at 200MHz means the HyperBus will run at 100MHz (DDR).

- The Quad-SPI (QPI) interface can run at up to 50MHz in SDR (single data rate) mode, with 4 data bits per cycle. The QPI clock rate is programmable in sub-multiples of half the FC clock speed (i.e. QPI Clock Speed = (Fclk_fc/2)/N, limited to 50MHz maximum).

- The Quad-SPI interface therefore offers lower bandwidth than HyperBus – however, the actual difference is mitigated by the fact that HyperBus is not much more efficient when transfering small chunks of data, due to latency cycles an other overheads.

- QPI memories typically come at a significantly lower price and may be easier to source.

- Cypress provides HyperBus RAM and Flash memories, as well as HyperBus Flash+RAM single chip (Multi-Chip Modules, MCM) – for example, the S71KS512SC0 MCM provides 64Mbit of RAM and 512MBit of Flash. HyperFlash is also available from a few other manufacturers such as ISSI, but HyperRAM and Hyper Flash+RAM MCM are available from Cypress only at the time of writing.

- QPI Flash are available from many vendors in a wide variety of capacities.

- QPI (pseudo-static) RAM chips are available from e.g. APMemory or LyonTek, in capacities up to 64Mbit at the time of writing.

6.2 - Specific Precautions when Interfacing with Quad-SPI Memories

Some QPI memories (typically, those housed in an 8-pin package) multiplex a HOLD#/RESET# signal on pin SDIO3. **When using such memories, it is necessary to connect SDIO3 to a weak pull-up resistor (e.g. 50K-100K) to SPIM_VDDIO**. Refer to AN002 [*Ref.5]* for details and rationale.

6.3 - PCB Placement and Layout Recommendations for external Memories

When interfacing with a HyperBus memory, clock rate could be up to 125MHz maximum (= 0.5x maximum FC clock frequency) ; when interfacing with a [Quad-] SPI memory, it could be up to 50-60MHz.
Although not extremely high, these are respectable speeds and careful layout of the clock and data lines is required to preserve signal integrity. This includes keeping trace lengths relatively short and balanced, avoiding as far as possible – or at least limiting – the number of vias on those traces, etc.
Cypress, the main provider of HyperBus memories, has produced an Application Note AN 211622 *[Ref.2]* « HyperFlash and HyperRAM Layout Guide » that provides multiple guidelines. Although they are somewhat on the conservative side, especially with regard to signal length balancing, they are a good starting point. It is recommended to follow them as much as possible, weighting them against practical constraints.
QPI Memories run at lower frequencies but PCB design, although a bit less critical, remains important. In addition, experience has shown that proper decoupling of power supply pins (with at least 1uF very close to the VCC pin) is essential.

7. JITTER ON GENERATED CLOCKS

Under certain conditions, GAP8's peripheral clocks can exhibit noticeable jitter. Acceptable jitter is application dependent. The root cause of this jitter is noise on the core power supply generated by the internal DC-DC. Application Note AN001 *[Ref.4]* provides jitter measurements and suggest some measures to mitigate this effect.

In most cases, jitter can be minimized by software, adequately programming the internal FLL and the DC-DC converter.

However, in some cases (if jitter needs to be minimized while running the DC-DC at very light load with minimal power consumption), an additional measure consists of filtering the low frequency noise present on the signal out of the DC-DC converter, using an RC filter with very small R (e.g. 0.3ohm) and large C (several tens of uF). It is therefore suggested to provision this RC filter at least on the initial vesion of the board (as shown in reference schematics, Fig.1), then depending on board requirement and constraints it may make sense to populate/adjust the R and C components.

8. SHARED GPIOS

GPIOA0 through GPIOA5 are available each on two different pins of GAP8, as indicated in section 4, « Pin-out and Pin Description » of GAP8's datasheet *[Ref.1]*. For instance, GPIOA0 is available as « alternate function 1 » both on pin A4 and on pin A3.
Therefore, take care, when desiging your system, not to inadvertantly use the same GPIO on different pins for different purposes. For example, if pin A4 is configured as GPIOA0 (i.e. set as alternate function 1), then GPIOA0 should not be simultaneously enabled on pin A3 – which in this case can be used in any other mode than Alternate 1.

9. JTAG

Any GAP8 board needs to provide access to GAP8's JTAG interface, as GAP8 programming and debug is performed through JTAG (except, possibly, in a production context, if the boot Flash attached to GAP8 gets pre-programmed stand-alone before essembly). The JTAG interface on GAP8 consists of pins JTAG_TDI, TDO, TMS, TCK and NTRST (optional). The JTAG programming probe will also require access to the harwdare reset pin NRESET. Most JTAG probes will also make use of a VDD pin (used to detect presence and/or voltage level of connected I/Os). Refer to Fig.1.
> NRESET : the JTAG probe needs to be able to pull the reset line low. When not asserting reset, the probe will not drive NRESET, which would typically be pulled to SAFE_VDDIO on the board through a resistor of a few 10Kohm.
> JTAG_NTRST : some JTAG probes (or probe adapters) make use of the JTAG_NTRST, others don't (beware, some simply pull JTAG_NTRST low which means it's permanently asserted). It is quite possible to program GAP8 without using JTAG_NTRST. The only drawback is <TBD>.
If JTAG_NTRST is not used, this input pin can be left open, as GAP8 integrates a weak pull-up resistor that deasserts JTAG_NTRST when not driven.
To accomocate all scenarii, it is suggested to **bring JTAG_NTRST to the JTAG connector through a series resistor** that would not be populated on the PCB if the probe (or probe adapter) does not use JTAG_NTRST, but be populated if the probe properly drives JTAG_NTRST.

In addition, it is recommended to insert a series resistor of a few 100s ohm on each JTAG line to protect it from unintended over-voltage when manipulating/connecting the JTAG probe and to limit ringing (as external wires may be of significant length).

10. UART FLOW CONTROL

The UART of GAP8 does not natively support hardware flow control (CTS/RTS). In addition, GAP8's UART does not buffer more than 1 byte – although a « software » FIFO can be implemented using the uDMA when messages received over UART are of known, fixed length.

When received messages are of unpredictable length and a baud rate higher than the standard 9600 baud is required, some means to throttle the data trafic (i.e. implement a h/w flow control solution) may be required. To that aim, GAP8's SDK supports « emulated » hardware flow control. **This solution requires to dedicate 3 GAP8 pins to UART flow control, of which one must support Timer functionality.**
Connections would be as depicted in Figure 5.

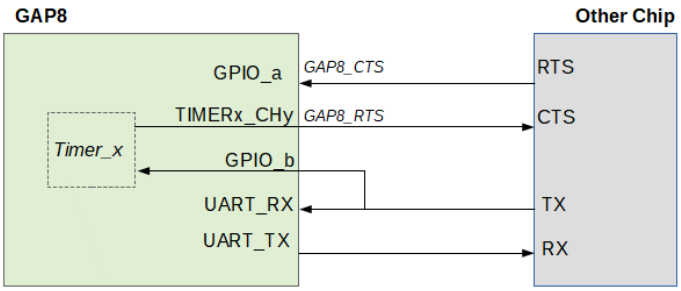


Fig. 5 : GAP8 UART hardware flow control connections

Besides normal connections of UART_TX and UART_RX :
- one pin (configured as GPIO, named GPIO_a on drawing) behaves as GAP8_CTS and connects to RTS of its counterpart,
- one pin (configured as timer output, TIMERx_Chy on drawing) behaves as GAP8_RTS and connects to CTS of its counterpart,
- one pin (configured as GPIO and internally routed to TIMERx input, GPIO_b on drawing) connects to same signal as GAP8's UART_RX.

GAP8's SDK lets the user select which 3 pins he/she would like to dedicate to that purpose. Just make sure 2 of them are GPIO capable and the third one is usable as Timer output.

11. KNOWN LIMITATIONS ON I/O USAGES

At the time of writing this Application Note, under certain conditions, a couple of I/Os of GAP8 cannot be used with full intended flexibility. *This will be fixed in an upcoming revision of the chip.* The affected I/Os are the following.

> Exclusive usage of UART Rx and HyperBus

When pin B7 is set in a configuration other than its default (Alternate-0), then the UART_RX signal that can normally be received through pin B6 is internally forced to '1'.
Since HYPER_DQ|6] is the Alternate-3 functionality of B7, this means **it is not possible to simultaneously use the HyperBus interface (HyperFlash / HyperRAM memories) and the UART RX functionality.**
A work-around is, when possible, to temporarily set back B6 to its default functionality (therefore preventing usage of HyperBus) during the time UART RX needs to be used.
NOTE :
- **UART TX** is not affected and **can always be used normally**.
- Also, **UART_RX can be normally used when using [Quad-]SPI** rather than HyperBus as interface with external memories.

> Exclusive usage of VSYNC and alternative functions of pin B34

When pin B34 is set in any configuration **other than its default** (Alternate-0 = I2C1_SDA), then the CAM_VSYNC signal that can normally be received through pin A36 is internally forced to '1'.
Therefore, **to maintain the integrity of VSYNC, do not use B34 as either GPIO15 (Alt-1 functionality) or TIMER3_CH3 (Alt-2 functionality)**. Using pin B34 as I2C1_SDA (its default functionality) is perfectly fine.

CAM_VSYNC is typically required in applications that connect an image sensor to GAP8. Applications that do not require usage of the VSYNC frame synchronization signal are therefore not affected by this limitation.

12. OTHER LAYOUT RECOMMENDATIONS

Beyond the usual good practices – keeping decoupling caps as close as possible to the pins they bypass, using polygons or wide traces for power supplies, as much as possible using a ground plane, etc. – it is recommended to pay attentions to a few specific areas when designing a board around GAP8.

In particular, note that GAP8's pinout is such that I/Os pertaining to the DC-DC converter – which is a fairly noisy block – neighbour I/Os pertaining to the 32KHz crystal oscillator – which is a noise-sensitive block. It is therefore essential to pay special attention to PCB design in these 2 areas, as highlighted in the corresponding chapters further up this document.

Precautions regarding the interfacing with external memory chips, which would often be the fastest signals around GAP8, were also highlighted earlier in the chapter on external memories.

Finally, here are a few additional hints and suggestions regarding GAP8 footprint and trace fanout :

GAP8 is packaged in an 88 pin, dual row QFN package (AQFN-88). See GAP8 datasheet, section 9 *[Ref.1]*. Fig. 6 below provides the same information in a slightly different format. Note this shows the dimensions of pads on the package, not landing pads on PCB.

The exposed central pad underneath the package should connect to GND through multiple vias, as, internally to the package, all GND connections of the die are through this pad. As is often the case with large copper areas, it is recommended to design the paste layer so that stencil aperture only covers about 75-80% of the exposed pad area – preferably doing so with a hatch pattern, i.e. keeping some spacing between paste zones (for example, 0.2mm).

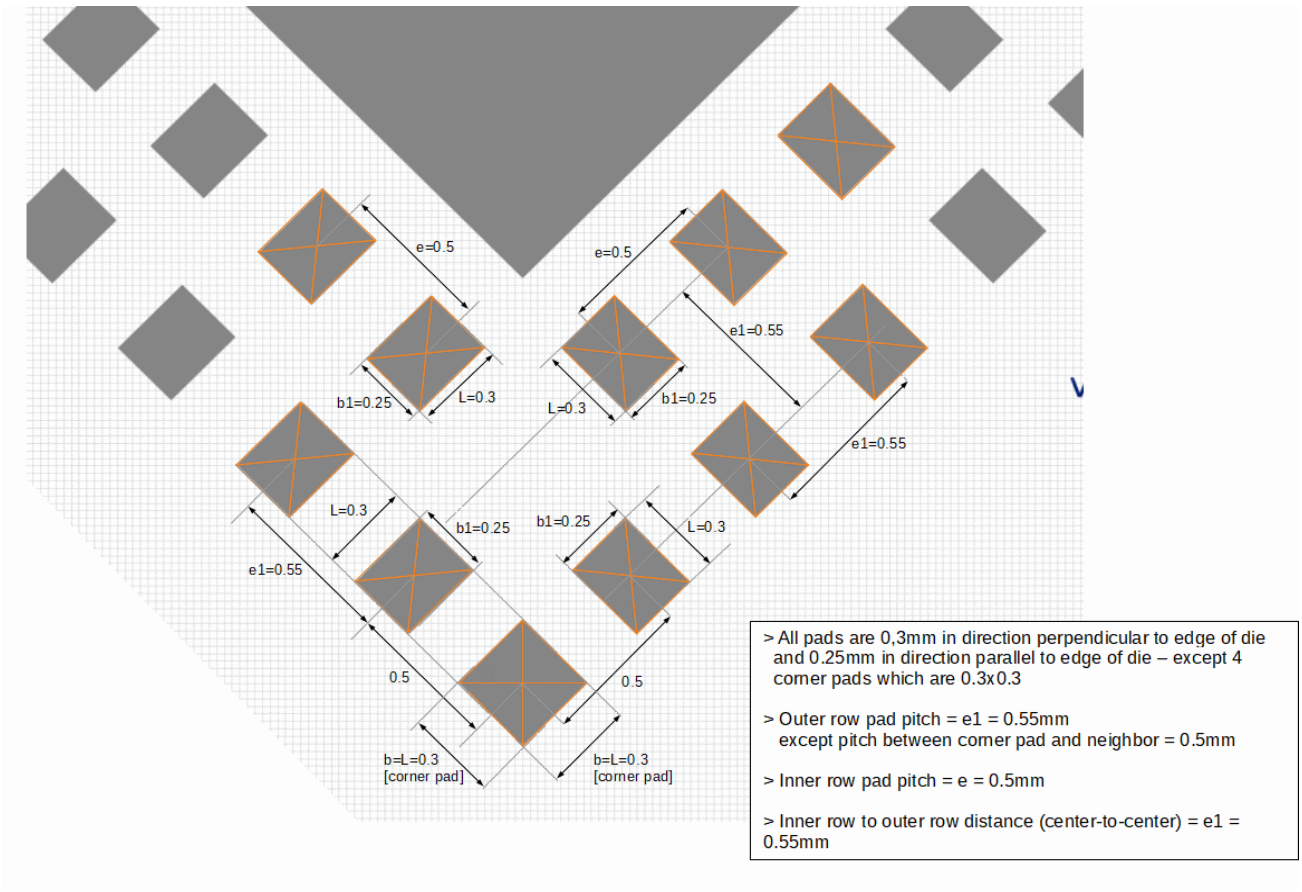
Two strategies are possible to fanout signals from the inner row.

> The land pattern recommended by the AQFN package manufacturer is described in the datasheet section referenced above : each landing pad is 0.3mm wide (0.35mm for corner pads). The minimum pitch between balls being 0.5mm, this leaves only 0.2mm clearance between landing pads on the PCB. This means it's highly impractical to fan out traces from the inner row between landing pads of the external row. With the central exposed pad also blocking traces from the inner row, in most cases it will be necessary to adopt a via-in-pad solution (and therefore resin filled vias) to fan out signals from the inner row.

> In some cases it may be interesting to consider an alternative solution, departing a bit from this land pattern and making each landing pad (of the outer row at least) the same width as its GAP8 pad counterpart. That leaves 0.3 mm clearance between landing pads of the outer row, which is enough to fit a signal trace using readily available technology (for example, 0.1mm trace with 0.1mm clearance on each side). Then it is possible to fan out all traces from the inner row staying on top PCB layer, removing the need for via-in-pad.

In any case, it is highly recommended to get advice from the PCBA shop that will handle assembly.

Fig. 6 : Reminder – GAP8 Physical Information (package pins)



REFERENCES

[1] GAP8 Datasheet and Reference Manual

https://greenwaves-technologies.com/sdk-manuals/

|2] Cypress AN211622 « HyperFlash and HyperRAM Layout Guide »

https://www.cypress.com/file/278156/download

|3] HyperBus Specification

https://www.cypress.com/file/213356/download

|4] GreenWaves Technologies, Application Note AN001 : « Minimizing jitter on clocks generated by the GAP8 application processor »
https://greenwaves-technologies.com/sdk-manuals/

|5] GreenWaves Technologies, Application Note AN002 : « Precautions required to use Quad-SPI Flash Memory with GAP8 »
https://greenwaves-technologies.com/sdk-manuals/

DOCUMENT HISTORY

Rel. 1.0, Feb-2020 (XC) :
Initial release

Rel. 1.0.a, Feb-2020 (XC) :
- Fixed reference to datasheet section in layout recommendations around GAP8 DC-DC converter
- In section 11, about Vsync/B34 mutual exclusions, clarify that I2C is available on other pins than B34

Rel. 1.1, Feb-2020 (XC) :
- Minor update to text about jitter on generated clocks