GAPMod 1.x

DOCUMENTATION

Rel. 1.1
5-Jun-2019

CONTENTS

Disclaimer

This information is preliminary and subject to change.

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

GAPMod is a core module architectured around GreenWaves Technologies' GAP8 chip, an ultra-low power Application Processor for the IoT (*Ref[1]).*
GAPMod is intended to facilitate and speed up development of applications around GAP8.
It packs on a small board (*ca.* 26mmx36mm) the "invariant" part of any application based on GAP8 (GAP8 chip, external Flash+RAM memory, associated decoupling caps, crystal oscillator, passives for internal DC-DC, etc.). This core hardware happens to also be, in many use cases, the most demanding part of the overall design in terms of precautions to take and in terms of required PCB technology.

With GAPMod, an application PCB designer can simply drop the module in his/her design and rely on comparatively relaxed Design Rules and reasonably priced PCB technology for his/her own application-specific motherboard.

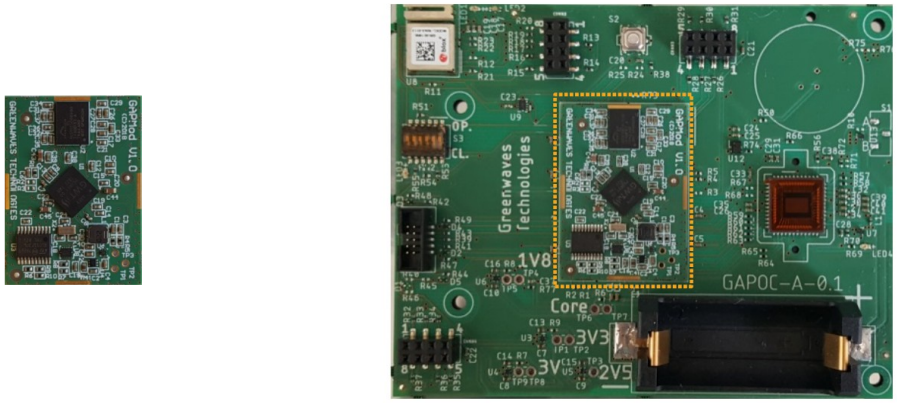

Fig. 1. : GAPMod module, stand-alone (left) and assembled on an application board (right)

2. Board Description

GAPMod comes in the form of a double-sided PCB to be assembled using SMT (Surface Mount Technology) on a target board. All components therefore sit on one side while the other side bears the SMT pads.

All GAP8 I/Os that are not dedicated to interconnecting chips present on GAPMod are made available to the target application board as GAPMod I/Os. For maximum flexibility, power supplies are also to be provided by the application board, with the exception of that dedicated to the on-chip crystal oscillator.
Figure 2 highlights the major parts that constitute GAPMod, while Figure 3 details physical dimensions of the board and location plus size of its I/O pads.

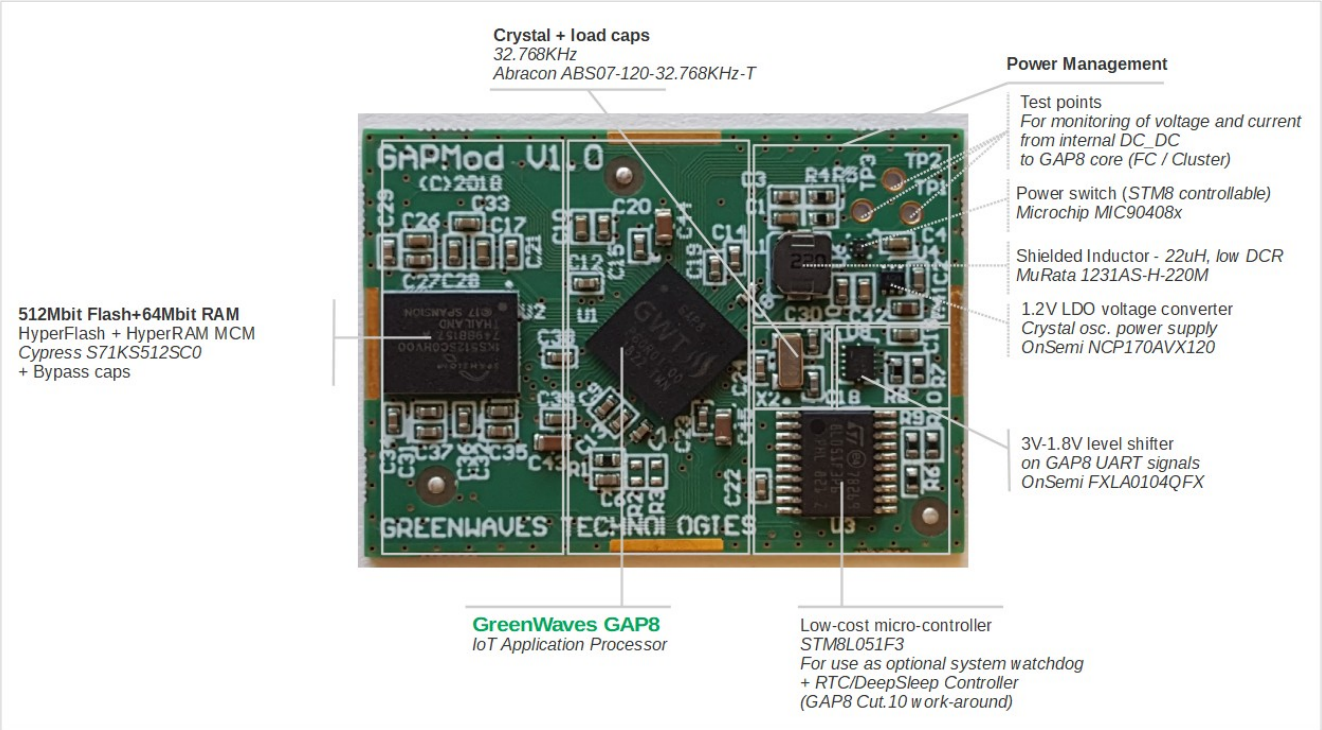

Fig. 2: Anatomy of GAPMod

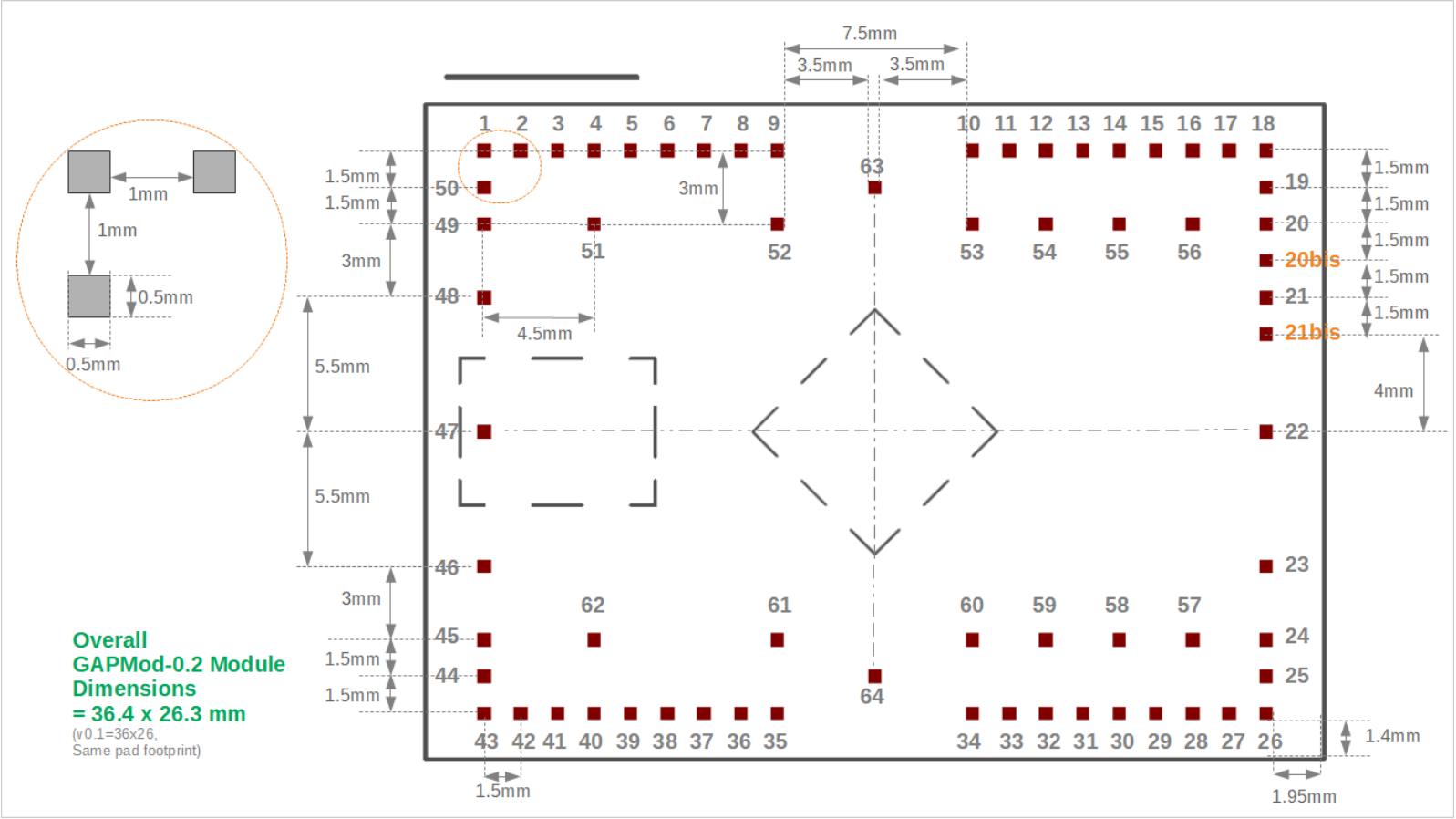

Fig. 3: GAPMod Physical Information, TOP (see-through) View
[Pads 20bis and 21bis exist on GAPMod1.1 only]
[Beware: some dimensions show a pitch (center-to-center), others show a clearance (edge-to-edge)]

3. GAP8

Central to the board is the GAP8 chip, an ultra-low power Application Processor for the IoT, bringing intelligence at network edge. GAP8 brings the ability to analyze and classify data from sensors (images, sounds, vibrations etc.) at very low cost and very low power, by extracting from sensor data condensed, meaningful information which can then be sent efficiently over a network.
GAP8 consists on an MCU-like part (a core plus standard peripherals), a processing intensive part (8 identical parallel cores plus a hardware accelerator specialized in Convolutional Neural Network operations), with on-chip L1 and L2 RAM and a specialized DMA. The cores are 32-bit RISC-V enhanced with specialized instructions.
Refer to the GAP8 Product Brief and Datasheet *(Ref.1]) for details.*

4. On-board power management

Refer to Fig. 3 for a graphical illustration of power management on GAPMod.
As mentioned earlier, most power supplies are to be provided by the application board in the sake of flexibility.
However, a 1.2V power supply derived from Vroot (typically battery voltage) is generated on-chip and dedicated to powering the on-chip crystal oscillator. This is to ensure the oscillator, which is quite noise sensitive, always works off a clean and stable power source. *[In future revisions of GAPMod, it is envisaged to power the crystal oscillator directly off VREG, the regulated voltage generated by the internal DC-DC converter – once it has been confirmed this does not degrade performance of the oscillator].*
Note also that power supply input “3V_1V8_MEMCORE_MEMIO” goes to the on-board Hyperbus memory and associated I/Os of GAPMod, so its value is dictated by the requirement of this memory chip, i.e. 1.8V (nominal) as GAPMod1.x is normally populated with a Cypress HyperFlash+HyperRAM memory chip in its 1.8V flavor (1.7V min, 1.95V max). However, should there be variants of GAPMod that employ the 3V memory flavor (2.7V min, 3.6V max), then the application board would need to provide 3V rather than 1.8V on input 3V_1V8_MEMCORE_MEMIO.

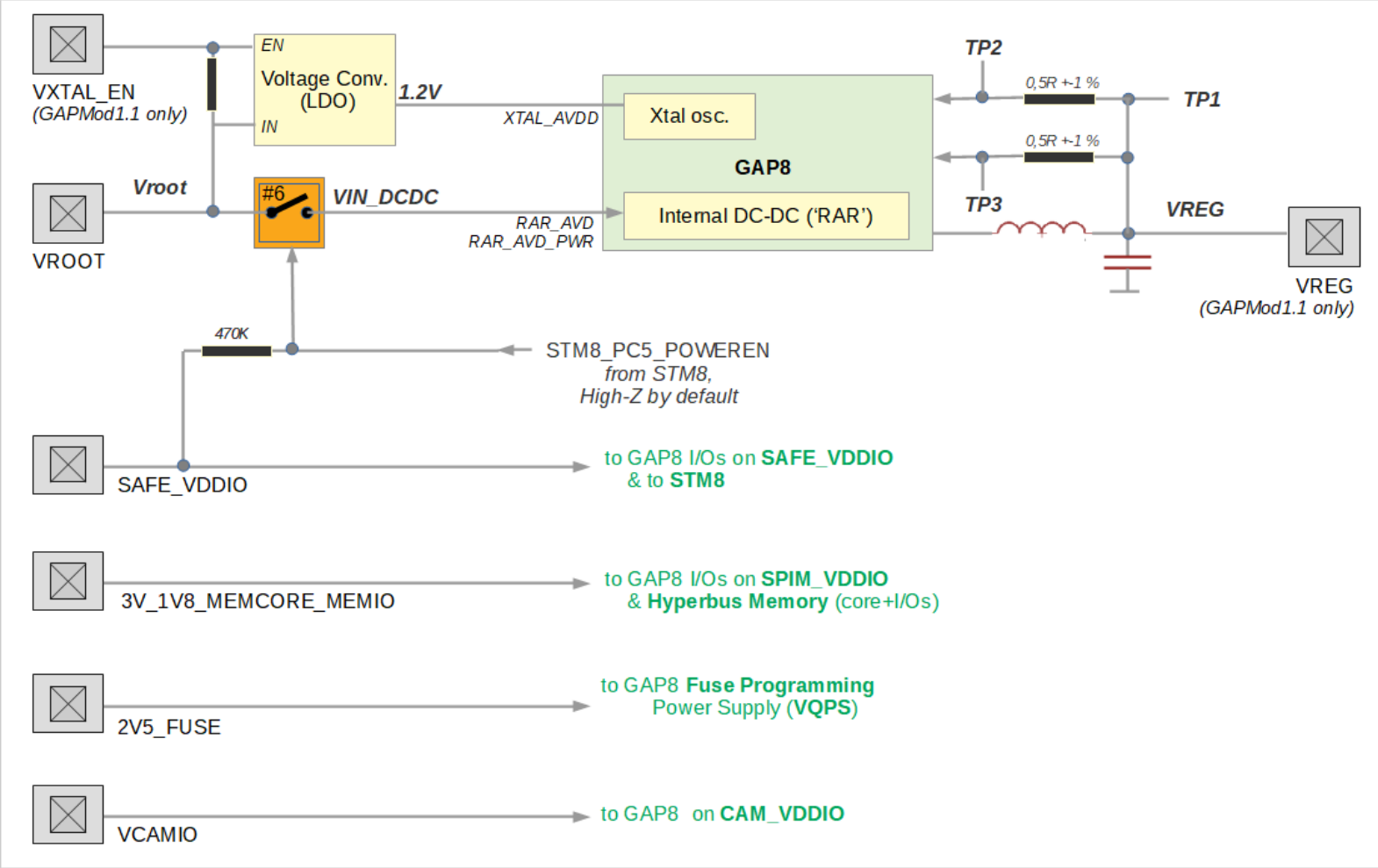

Fig. 3 – On-board power supply generation and control (notional)

– Test Points / Power Supply Monitoring –

3 testpoints are provided to monitor supply voltage and supply currents drawn by the "SoC" part (a.k.a FC, Fabric Controller) of the GAP8 chip and by the "Cluster" part.

> A 0.5 ohm (+-1%) sense resistor placed on the current path **between TP1 and TP2** allows to measure the **current consumption of the SOC region**.

> A 0.5 ohm (+-1%) sense resistor placed on the current path **between TP1 and TP3** allows to measure the **current consumption of the Cluster region**.

The current (in mA) is twice (2x the value of the voltage drop (in mV) measured across the sense resistor (I=U/R). At 0.5ohm, the sense resistor is deliberately small to avoid severe supply voltage fluctuations when there are significant instantaneous current changes.

These currents are measured at output of the DC-DC converter. The efficiency of GAP8's internal DC-DC converter will determine how this translate as actual power drawn by GAP8 for its core (SOC+Cluster).
Note: The efficiency of the DC-DC conversion is impacted by the DC resistance (DCR) of the inductance placed at its output – but lower DCR inductance also means bigger inductance size. On GAPMod the selected inductance has about 300 mOhm maximum DCR, which is good while allowing to keep inductance size reasonable.
In addition, *on GAPMod V1.1 only :*
> the regulated voltage VREG is made available on an I/O pad of GAPMod. This allows monitoring VREG; it could be used by external hardware to detect if GAPMod is in Sleep mode without relying on another specific signal, as:
- in normal mode, VREG may range from 1.0V to 1.2V
- in sleep mode, the internal DC-DC switches to LDO mode and provides VREG=0.8V.
> the enable input of the 1.2V voltage regulator used to power the on-board crystal oscillator is made available on an I/O pad too. Some external hardware may want to pull down this signal to disable the LDO thereby reducing its consumption to less than 0.5uA.
Also refer to GAPMod I/O list.

See also STM8 chapter below for note about how STM8 may impact measured currents, esp. in low consumption conditions.

– Software-controllable Power Switch –

A power switch placed on the current path feeding the internal DC-DC converter allows the on-board STM8 to cut this power supply, thereby completely switching off the core (FC+cluster) of GAP8.
This is mostly intended to work around the fact that GAP8 Cut1.0 has a weakness that limits its deep sleep current to a few tens of uA instead of about 1uA which is its true target (will be fixed on Cut1.1).
This also allows to use the STM8 as a system wide RTC and WatchDog.
This requires to program adequate firmware into the STM8, either through its SWIM programming interface or possibly through UART.
If STM8 is not programmed, it does not interfere and power to GAP8 internal DC-DC is always on. The switch is controlled through STM8's GPIO pin **PC5,** driving enable signal STM8_PC5_POWEREN in schematic. This enable signal is also made available as GAPMod I/Os and could be used by hardware external to GAPMod.

5. HyperBus Memory

GAPMod implements a Flash+RAM MCM (Multi-Chip Module), employing Cypress's HyperBus specification (*Ref[2])*. The selected part is Cypress's S71KS512SC0B, offering 512Mbit of Flash and 64Mbit of RAM and using 1.8V nominal power supply (1,7V min, 1,95V max).
It might happen that future versions of GAPMod employ different variants of Cypress HyperBus memory (e.g. Flash only, or 3V rather than 1.8V power supply, or different capacity). These are pinout-compatible and GAPMod's architecture allows to perform with swap without any other board hardware change.

6.Auxiliary STM8

V1.x of the GAPMod core module includes a small STM8 micro-controller (STM8L051F3) able to shut off GAP8, usable as RTC and as System WatchDog (see "on-board power management", above). It is used mostly as a work-around to overcome a limitation of GAP8 Cut1.0 (excessive power consumption in deep sleep) and will be removed in future versions of GAPMod, further optimized and designed to work with next revisions of GAP8.
In the meantime, GAPMod makes some STM8 I/Os available to the external world. Refer to the Functional Definition of GAPMod 1.x I/Os (*Ref[4]*).

NOTES:

1) Behavior of blank (non-programmed) STM8
The STM8 receives the system-wide reset NRESET on its reset input. A peculiar behavior of the STM8 worth keeping in mind is the following: a blank (non-programmed) STM8 will, after power-up, perform a self-reset every second – but in doing so, it also drives low its reset input (which therefore is not a pure input). On GAPMod v0.1 this causes a full system reset which is a problem – therefore some code, even dummy, should first be programmed in STM8 internal Flash (for instance, a halt() sequence). On later GAPMod versions, this issue has been fixed (by inserting a series diode on the STM8 reset input path, so that STM8 pulling its reset input does not impact the system reset).
However, if aiming at super-low consumption, it is still recommended to program the STM8 so that it is halted when no activity is required from it, otherwise it'll be running an infinite loop and therefore consume a few hundreds uA or more.

2) GAPMod pin STM8_PB0_ADC going to STM8
This pin connects through a to an STM8 I/O that may be used as ADC input. With adequate firmware flashed into the STM8; this enable measurement of some voltage level, perhaps through a resistor bridge whose foot could be enabled (grounded) or disabled (high-Z) using pin STM8_PD0/GNDPULL.
However:
1) pay attention to the fact that the STM8 won't reliably sustain a voltage greater than 3.6V on its I/Os

2) also pay attention to the fact that putting on this STM8 I/O pin a voltage greater than STM8's power supply voltage plus a few hundreds of mV will trigger its internal I/O protection diodes and therefore cause current to be drawn.

7. Programming / Debugging

> GAP8 :
Code can be downloaded into GAP8 internal RAM or into the Hyperbus Memory present on GAPMod through its JTAG interface. This is done using the **standard GAP8 SDK**, please refer to its documentation *[Ref.3].*
> STM8 :
The STM8 present on GAPMod can be programmed using its standard SWIM interface, consisting of: power supply, Ground, Reset (NRESET) and single-wire data interface (SWIM). Refer to STM8 documentation for details. These signals are made available on GAPMod I/Os as specified in the GAPMod I/O Table (see below).

8. GAPMod I/O Signal Assignements

Please refer to Table 1 "GAPMod I/Os" (provided as a separate spreadsheet for now (*[Ref.4]*).

REFERENCES

Ref[1] -
Product Brief "*GAP8 IoT Application Processor*" – available from *greenwaves-technologies.com*
DataSheet "*GAP8 Hardware Reference Manual*" – available from *greenwaves-technologies.co*m

Ref[2] -
"HyperBus specification – Low signal Count, High Performance DDR Bus"
"HyperFlash and HyperRAM Multi-Chip Package 1.8V/3V"
www.cypress.com/products/hyperbus-memory

Ref[3] -
GreenWaves GAP8 SDK & Manuals : *greenwaves-technologies.com/en/sdk/*

Ref[4] -
Functional Definition of GAPMod 1.x I/Os :
Spreadsheet *GAPMod_v1.x_IO_Definition.xlsx* available from GreenWaves
Pin # on GAPMod vs. connection of GAPMod (e.g. Pin ID on GAP8 if relevant) and brief description of intended usages.

DOCUMENT HISTORY

Draft A – Dec.2018 (XC)
Initial Draft

Draft B – 17-Jan-2019 (XC)
Added Disclaimer & License type

Rel. 1.0 – 23-Apr-2019 (XC)
Changed marking from Draft to Rel.1.0

Rel. 1.1 – 5-Jun-2019 (XC)
Added note in STM8 section about current consumption.