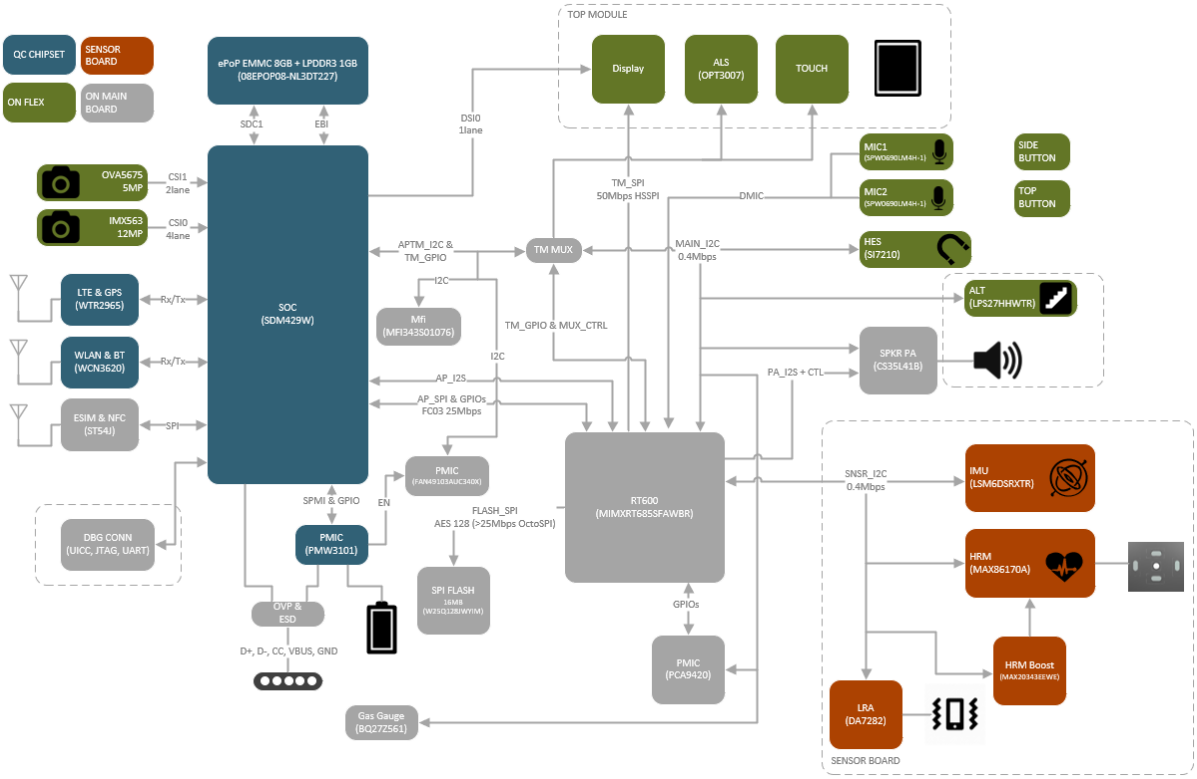
WIP Big-Little Architecture

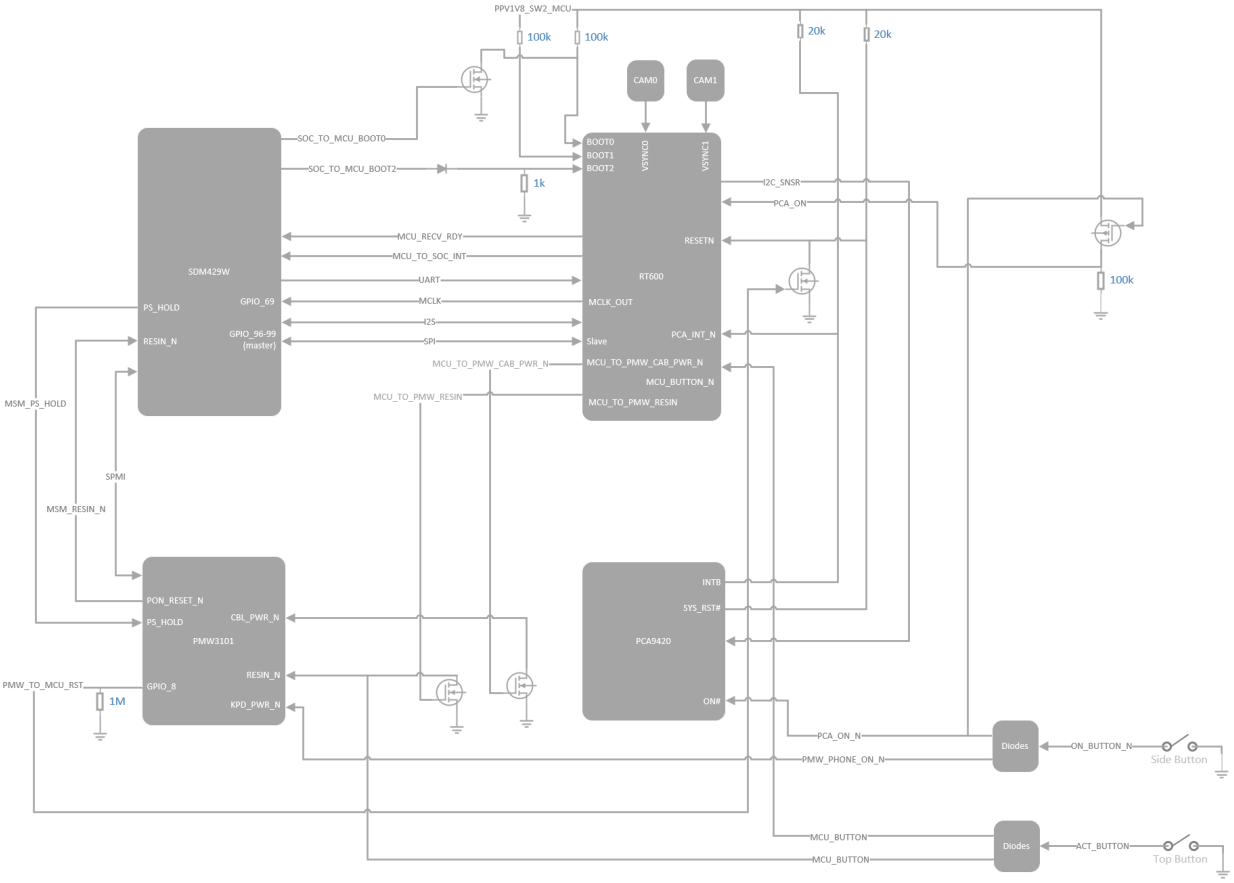
The Milan architecture involves the Big+Little, with SoC (big) focusing on CPU/GPU heavy activities and MCU (little) focusing on low compute but always on activities. The baseline candidate for MCU is RT600.

Block Diagram

The diagram below shows the interfaces between SoC, MCU and other subsystems.



Control/Handshake between SOC and MCU



Boot Up

SoC and MCU should both support boot via button press or VBUS (putting on charging dock) out of ship mode. During boot up MCU should default to external flash (SoC should configure the boot pins as highZ). MCU will wait for SoC to fully boot before comm link is established. (SPI interface handshake, ie: version number check).

- The MCU POR (Power On Reset) state must ensure the SOC has control of i2c and display signals.
- What happens in a dead battery: there will be first trickle charging, and then SoC will fully boot. There needs to be MCU management in this case since VPH will not be gated, where as SoC boot will be gated until the battery meets a certain threshold. (During trickle charging there is no option for charger animation as the display gets initialized in LK and trickle charging is part of SBL.)
- If SoC does not come up after x seconds, the MCU will display error message.
- SOC is responsible for displaying splash screen and boot animation (lk)

OTA Update

SoC has A/B partition. SoC will download and store the new image, then boot into the new partition first. SoC should be updated first, reboot and then check MCU version, if MCU's FW needs to be updated, it should allow a silent background update without requiring further reboot (This should be achievable via reset + boot pin configs + UART0). Image will be loaded to the MCU external flash first before boot.

- MCU FW image should have a single partition. Since the same complexity is required from SoC side regardless if A/B is included or not.
- There is no plan to independently update MCU FW without SoC updating as well.
- MCU is only planned to be updated during a SoC OTA update.

Display Handoff

Display mux control should default to SoC during bootup (external passive pull up), it is actively controlled by MCU otherwise. Panel by default boots to DSI mode, and requires a DSI command before allowing SPI control according to the datasheet. SoC should be in charge of display animation.

- TE signal should not be required from DSI interface, only needed for SPI drive.
- Display idle mode is 15Hz, touch is 20Hz, color depth should not be reduced per AUO.
- MCU GPIOs are Hi-Z on reset and POR.
- SOC DSI lines should be tri-stated when MCU is in control
- The component handed to is responsible for reconfiguring. MCU would be responsible for setting Display into IDLE, where the SOC would be responsible for setting it to Normal mode. This holds true for Touch and Haptics as well.

Data Transfer

The SPI interface is the main interface for data transfer between SoC and MCU (health sensor data, AOD display data etc.). There are two additional GPIOs between SoC and MCU, MCU_2_MSM_INT (for MCU to initiate SPI transfer) and MCU_RECV_RDY (for SPI out of band communication, used in Bluelink SM).

-Both pins are driven by MCU and are considered as interrupt inputs on the SoC side.

Fastboot Mode

During Fastboot mode, SoC enters Little kernel/bootloader, (MCU may be booted already) LK will have access to UART/SPI. FW update is applied in the next boot, same as in OTA. MCU will be held in reset once Little Kernel boots into fastboot mode.

- There is enable for MCU to enter ISP mode by script (using BOOT/RST pins)

MCU only Mode

MCU should turn SoC off via SoC SW turn off sequence.
MCU should be able to turn on SoC via CAB_PWR_N.
Can button (KPD_PWR_N) wake SoC during MCU only mode? Should there be a long hold requirement so that the SoC does not get accidentally turned on?

Outstanding Questions

Does the SOC need the TE signal, or is the DSI data enough?

-TE is a part of DSI protocol, it is not needed during DSI drive.

How can the SOC/MCU communicate this since STP takes over SPI for Bluelink? Can the UART be used for this?
How long does it take to transfer the MCU firmware?