Hardware Meeting

- **D10 Chip**

  The D10 chip is a unique Sivantos product and contains several analog and digital chips. The aim is to control the chipset remotely, e.g. simulate pushbutton press, analyse acoustic data generated on this chipset. The chipset is not provided so make-shift solutions are supposed to be used instead, *e.g. toggling a push button to simulate LED going on / off.*

- **Volume Control**

  DVC: Digital Volume Control

  AVC: Analog Volume Control

  Create Positive / Negative Pulses on these lines to increase / decrease Volume. On some hearing aids, there are rocker switches OR digital volume controls / endless volume control, either AVC or DVC is operated. `GPIO20, GPIO21` shouldn't be pulled HIGH together, it's undefined, possibly dangerous. One only operates it by increasing/decreasing volume.

- **DET_AUDIO, DET_TELE**

  Magnetic Switches detect proximity to Ear and switch between Phone / external Audio Sources. Connector also has a sensor Switch automatically activating when sources are connected.

- **GPIO Voltage Level is not compatible with Chipset, that's why we have analog switches.**

- **Multiplexer**

  For each multiplexer-> table in the program stored. Connect PUSH_BUTTON to corresponding User1-9 Pins. XML Configuration -> Which GPIO is used for which type of user controls.

  - X0 -> USER1 connection is always fixed.
  - Not all Pins are used, they're grounded.
  - Y2 -> LED -> IN2 (to DAC) is also always fixed.
  - PUSH BUTTON is always Y3.
  - But the connection in between needs to be rerouted.
  - Sticking to existing XML-file would be nice-to-have, but it can be slightly modified. The configuration embedded in the provided file is however valid.
  - Other Pins (not shown in circuit diagram) are not connected / used.
  - Reducing Voltage -> Low Voltage Melody heard -> Emulates Low Battery. ADC-DAC is linked to this somehow, adjusting OUT1.
  - During communication nRESET has to be pulled up to 1. If it's 0 there's undefined behaviour, the chip might not even connect / talk back.

- **MCP Potentiometer**

  VBAT, going all the way in the potentiometer might cause danger!

  1.5V should not be exceeded, the hearing aid can't handle more (pussy)

  Hearing Aid decrease output volume when voltage ijs as low as 1V and at some point, it plays a "LOW BATTERY" melody.

- **Receiver**
  Receivers are Loudspeakers.
  ARD = Automatic Receiver Detection.
  Providing information about which receiver is connected: left or right (actually it's just a simulation).

- **REC1, REC2**
  Differential lines -> transferred to single ended via a woofer amplifier, after it's referenced as BEEP signal and fed to the digital converter. Basically, you do all this to detect sound reactions from the chipset. For this functionality to work properly a SPI connection needs to be established. The signals for REC1/2 are coming from the D10 chipset.

- **Use Digital to Analog Converter for Battery Simulation!**

- **Try detecting connection melody through Analog to Digital Converter!**

- **Interrupter simulating a broken cable.**

- **LED**
  Some hearing aids are equipped with LEDs, which can be detected as well and thus run through the ADC as well. Low-Pass-Filter used to "slow" the signal to a DC current signal (?)

- **Other LEDS can be used for debugging, they're not part of the actual productive EvalBoard**
  GPIO23, 24 can be used for this, e.g. indicating "Board is Ready" or "Running Test".

- **LED & BEEP are the only signals coming from Chipset**
  For BEEP a solution is already provided, for LED detection a solution needs to be developed (?)

- **"Fooling" the hearing aid, calculating which voltage to provide to hearing aid, to make it think a receiver is connected (VBAT).**
  Rx will be provided from a Table by Sivantos, VBAT however needs to be adjusted according to a specific formula. No *adjustable resistors* are used to accomplish this task because they aren't perfectly doing their job, and the chipset doesn't have lots of tolerance towards floating voltages.