|  |  |  |
| --- | --- | --- |
| **Technische specificaties** | | |
| **#** | **MoSCoW** | **Omschrijving** |
| **T1** | **M** |  |
| T1.1 | M | Er wordt gebruik gemaakt van een printplaat die zonder solderen verbonden kan worden aan de RPI en PICO |
| **T2** | **M** |  |
| T2.1 | M | De RPI afmetingen worden overgenomen van de volgende template: |
| T2.2 | X | X |
| T2.3 | C | De HAT bevat een barreljack 5.2mm formaat voor voeding |
| T2.4 | M | De connector voor de RPI naar de HAT is een 2.54 pitch 1mm tht header (stacking header). De connector voor de PICO is een THT pin header. |
| **T3** |  |  |
| T3.1 | W | Voor de connector type worden screw terminals en/of 2.54 female headers gekozen. |
| T3.2 | W | Voor de audio apparatuur wordt een tulp connector gekozen |
| T3.3 | S | De extra pinnen worden via de connector typen gespecificeerd in T3.1 uitgebroken. |
| T3.4 | M | Datacommunicatie tussen de RPI en PICO vind plaats via een USB kabel van USBA poort van de RPI naar micro USB poort op de PICO. |
| **T4** |  |  |
| T4.1 | S | De RPI pico wordt gevoed vanuit de RPI USB poort én als back-up vanuit de PCB. Dit volgens onderstaande circuit uit de RPI2040 datasheet, waarbij V=5V: |

|  |  |  |
| --- | --- | --- |
| T4.2 | C | De RPI wordt gevoed vanuit de Raspberry PI officiële voeding voor de 5V 5A volledig vermogen modus, of vanuit de printplaat naar de GPIO pinnen gevoed te worden op 5V ?A max. |
| T4.3 | X | X |
| T4.4 | S | De voedingsheaders leveren de directe voeding door aan de uitgangspoorten. Een minimale vorm van bescherming in de vorm van een diode en zekering wordt toegepast volgens onderstaande circuit |
| **T5** |  |  |
| T5.1 | M | Een ws2812b Led wordt gebruikt als status indicator |
| T5.2 | S | Een 6x6x6mm smd tactile knop wordt gebruikt als een reset knop. |
| T5.3 | S | Zelfde knop als bij T5.2 wordt gebruikt. |
| **T6** |  |  |
| T6.1 | S | Er worden soldeer jumpers geïnstalleerd op plekken waar functionaliteit kan worden geschakeld tussen RPI5 en RPI3 en RPI4, Hierbij is de standaard uitgaande van de RPI5. |

\* verklaring gebruik micropython

Zowel C als Python is in het verleden gebruikt om microcontrollers te programmeren. C is sneller dan Python vanwege de architectuur, C moet gecompileerd worden, Python is een taal die wordt geïnterpreteerd (Captain\_Pumpkinhead, 2021). Python is trager, maar aangezien er geen zeer hoge snelheid vereisten worden verwacht bij dit systeem zijn de voordelen van Python voor ontwikkeling overtreffend. Python is het platform dat zowel in de PICO als op de RPI wordt ondersteunt. Aangezien er veel voorbeeldcode voor de RPI beschikbaar lijkt te zijn voornamelijk in de codetaal Python en dat ontwikkeltijden in handen werkt is er op dat moment voor Python gekozen. Ik heb eigenlijk te snel gekozen voor Python gekozen, achteraf gezien waren er meer mogelijkheden. In de toekomst kan er beter gekeken worden naar alternatieven. Zo is C draait C dichter bij de hardware en is daarom vaak sneller. Dit kan interessant zijn voor het gebruik van embedded vision op de RPI. Daarnaast was het debuggen van de RPI pico zeer lastig. De IDE die ik gebruikte gaf weinig tot geen inzicht in wat er zich in de RPI pico afspeeld. Werken met Printstatements was mijn enige vorm van inzicht vrijwel in de PICO. De PICO debug probe was wel een optie geweest met C. Ik zat te diep in mijn ontwikkeling om nu nog de keuze te maken om over te gaan op C. Dit had teveel vertraging opgeleverd naar vermoeden.

Notitie: micropython kan niet daadwerkelijk multithreading gebruiken, dit heeft te maken met Pythons  [Global Interpreter Lock](https://wiki.python.org/moin/GlobalInterpreterLock) (GIL). (Joshy, 2017), iets dat meerdere mensen zeggen op meerdere websites.

Volgende keer de software testen zonder MCU, dit lijkt makkelijk te gaan.