Test Stand for RFPI

Rafał Kotas, Wojciech Jałmużna, Bartosz Pękosławski, Grzegorz Jabłoński

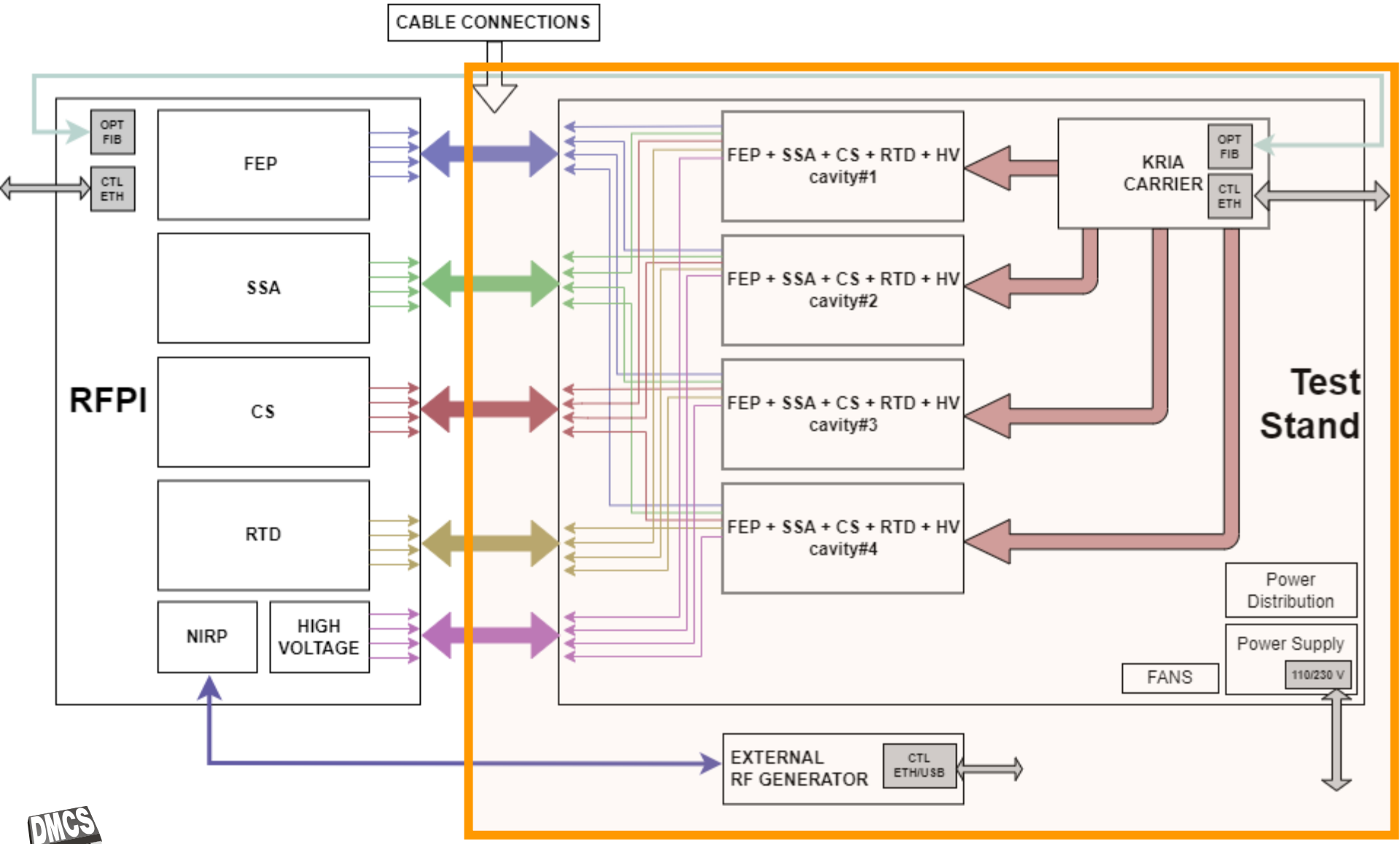
Agenda

- The concept
- Main requirements
- Test stand version specification and scope
- Functionality and design details
- Full scale design plans
- Summary

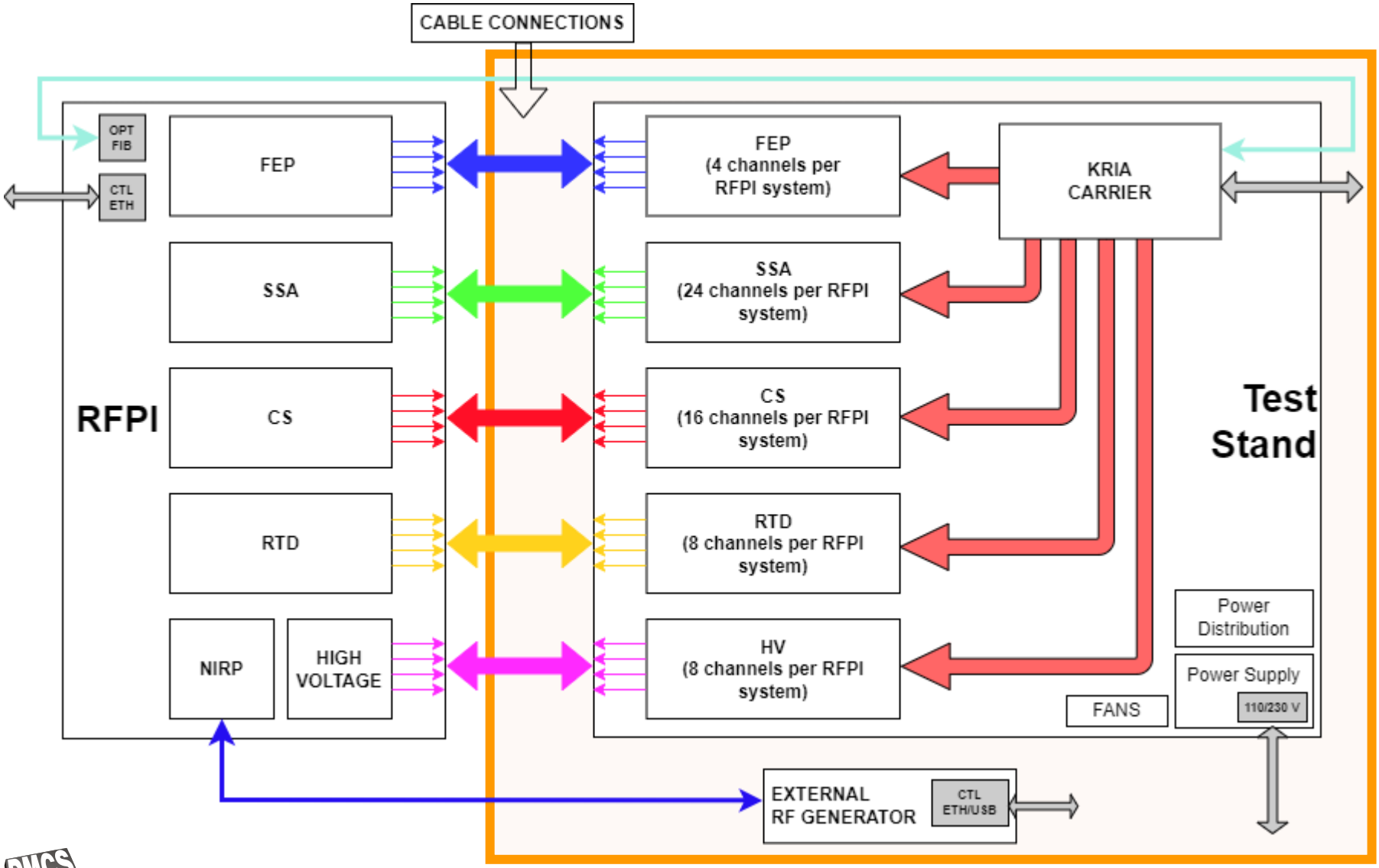
Product lifecycle

- 1) board assembly
- 2) board electrical tests
- 3) boards assembled into RFPI system
- **4) tests with test stand**
- 5) final evaluation
- 6) installation
- 7) post-deployment (comisioning)
- 8) maintaince
- 9) end-of-life decomissioning

The primary concept



The modified concept



The main requirements

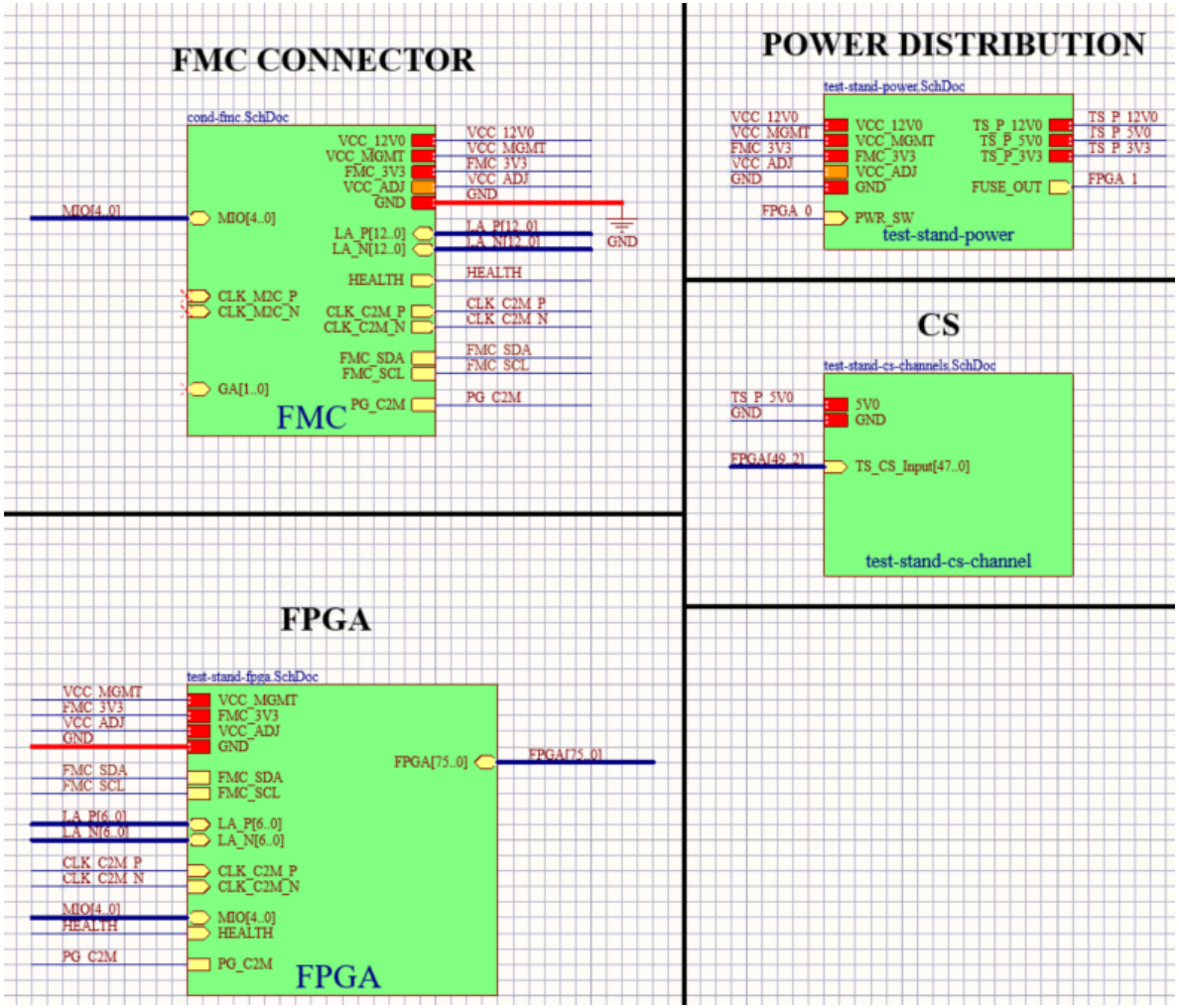
- The role of the test stand is to perform post-production testing of manufactured RFPI systems in automatic (semi-automatic) manner, including tests of:
  - CS conditioning boards
  - RTD conditioning boards
  - SSA conditioning boards
  - FEP conditioning boards
  - NIRP filters
  - High Voltage module
- The architecture of the system includes 5 boards
- EPICS interface to test stand

The test stand version specification and scope

- Chosen platform – Kria carrier board
- Operating system on the test stand – Ubuntu 22.04.1 LTS
- Design and implementation of a single test board per all types of conditioning boards
- ITX power supply providing 3.3 V, 5 V and 12 V separately

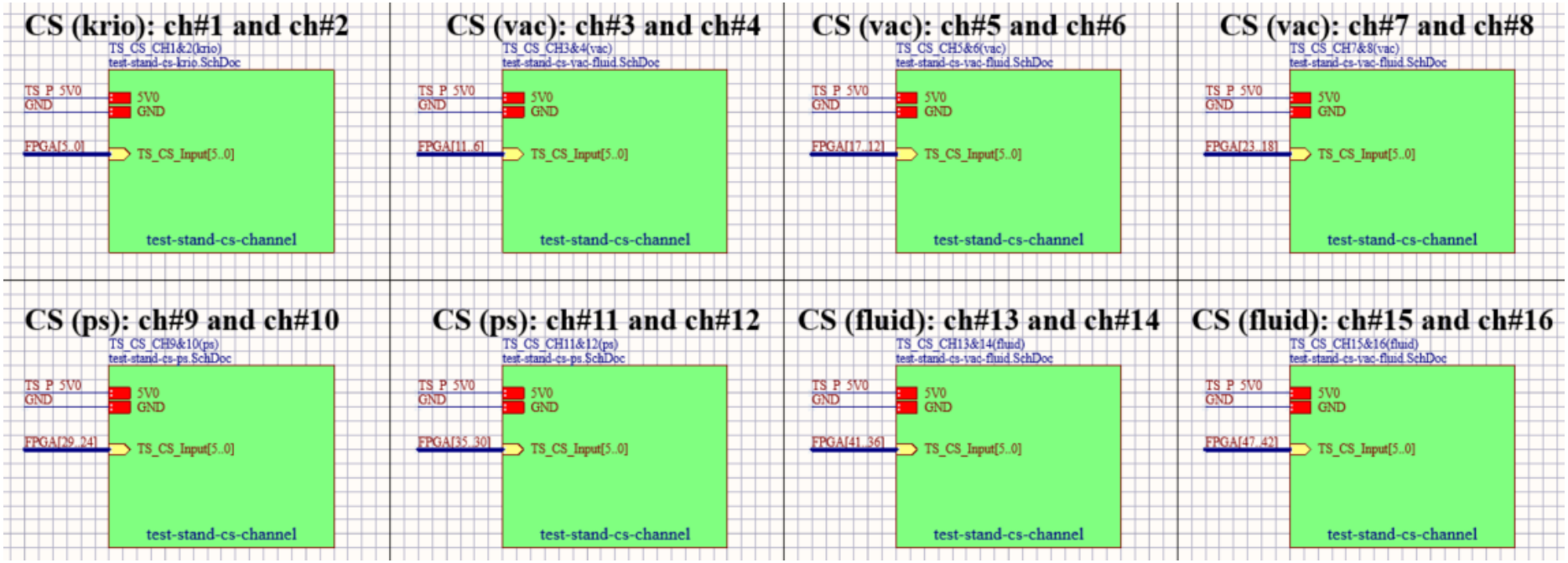
The functionality and design details

- The test stand board for CS conditioning boards (16 channels per RFPI system) – overall design



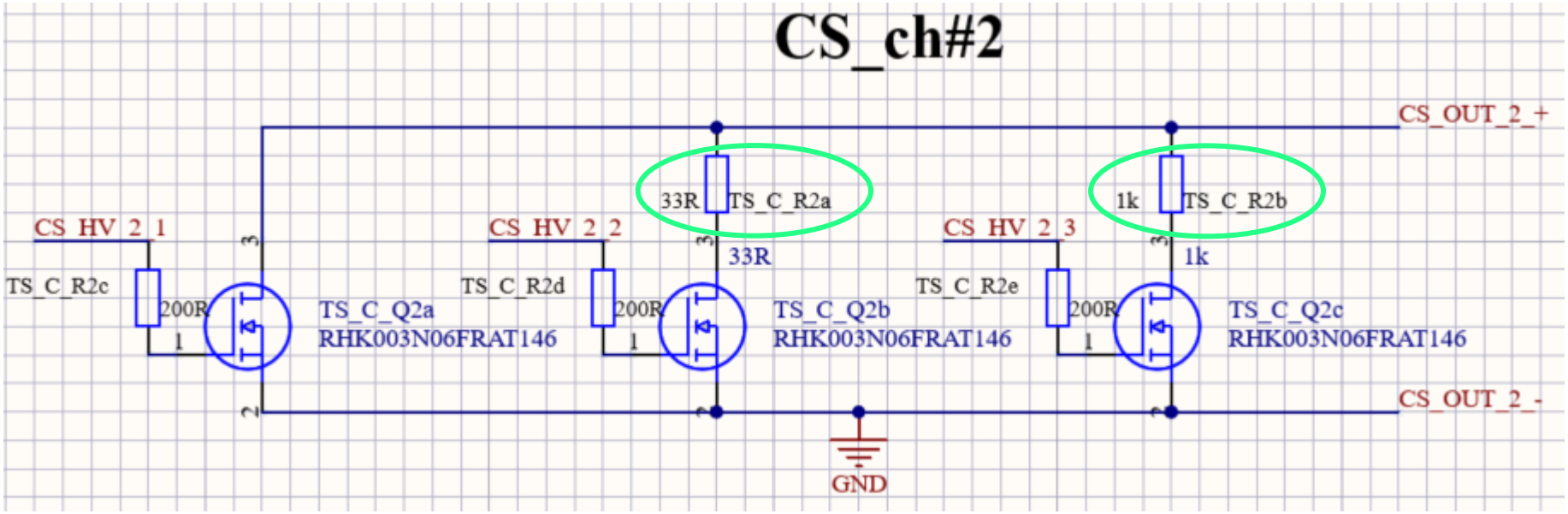
The functionality and design details

- The test stand board for CS conditioning boards (16 channels per RFPI system) – channel list



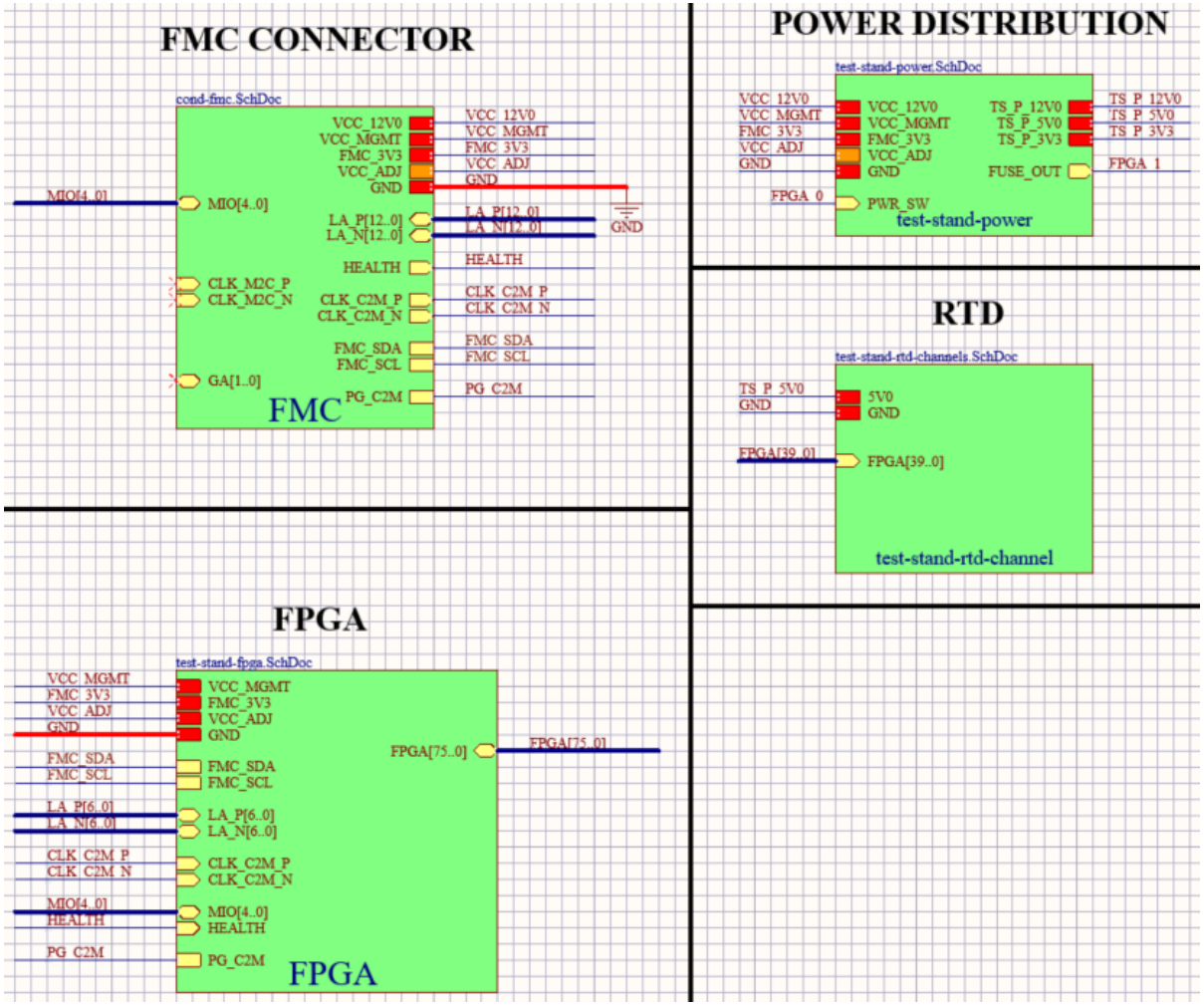
The functionality and design details

- The test stand board for CS conditioning boards (16 channels per RFPI system) – testing conditions:
  - normal operating conditions
  - short circuit
  - opened line
- Resistors differ by channel type



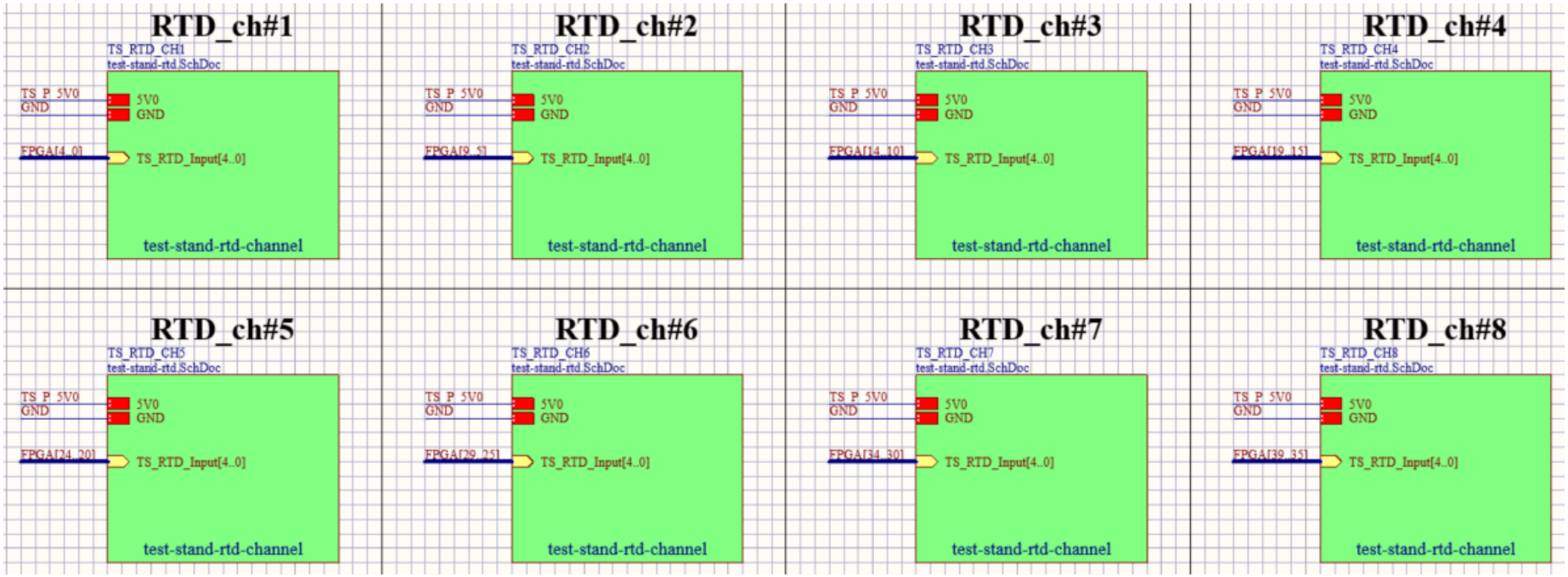
The functionality and design details

- The test stand board for RTD conditioning boards (2 channels per 1 cavity → 8 channels per RFPI system) – overall design



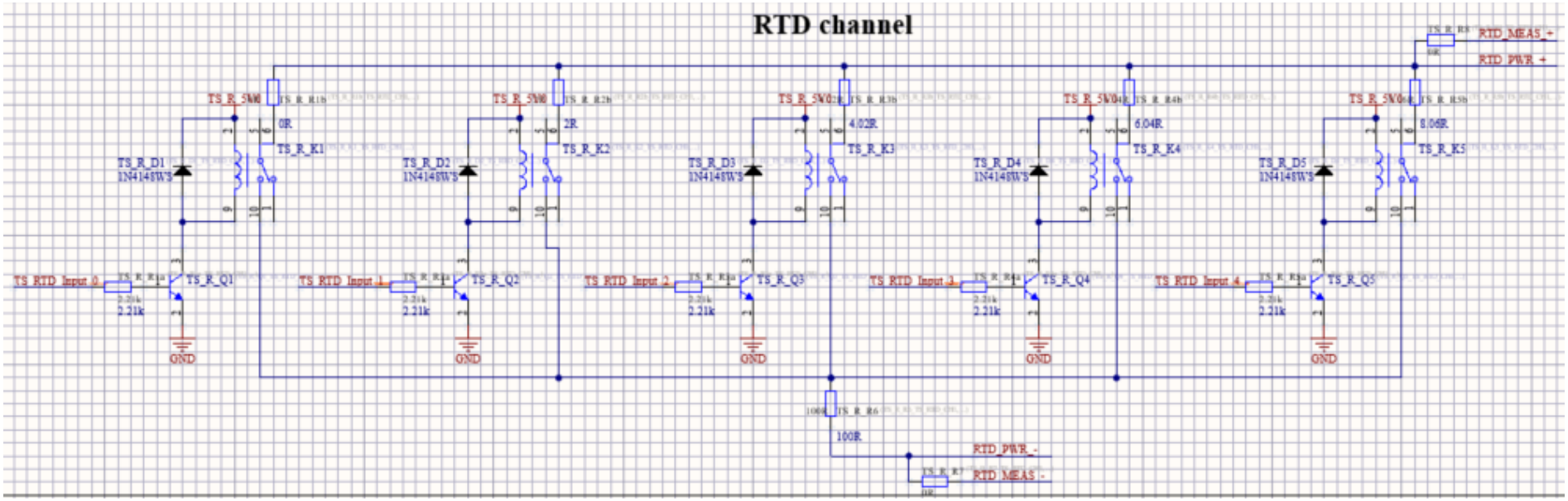
The functionality and design details

- The test stand board for RTD conditioning boards (2 channels per 1 cavity → 8 channels per RFPI system) – channel list



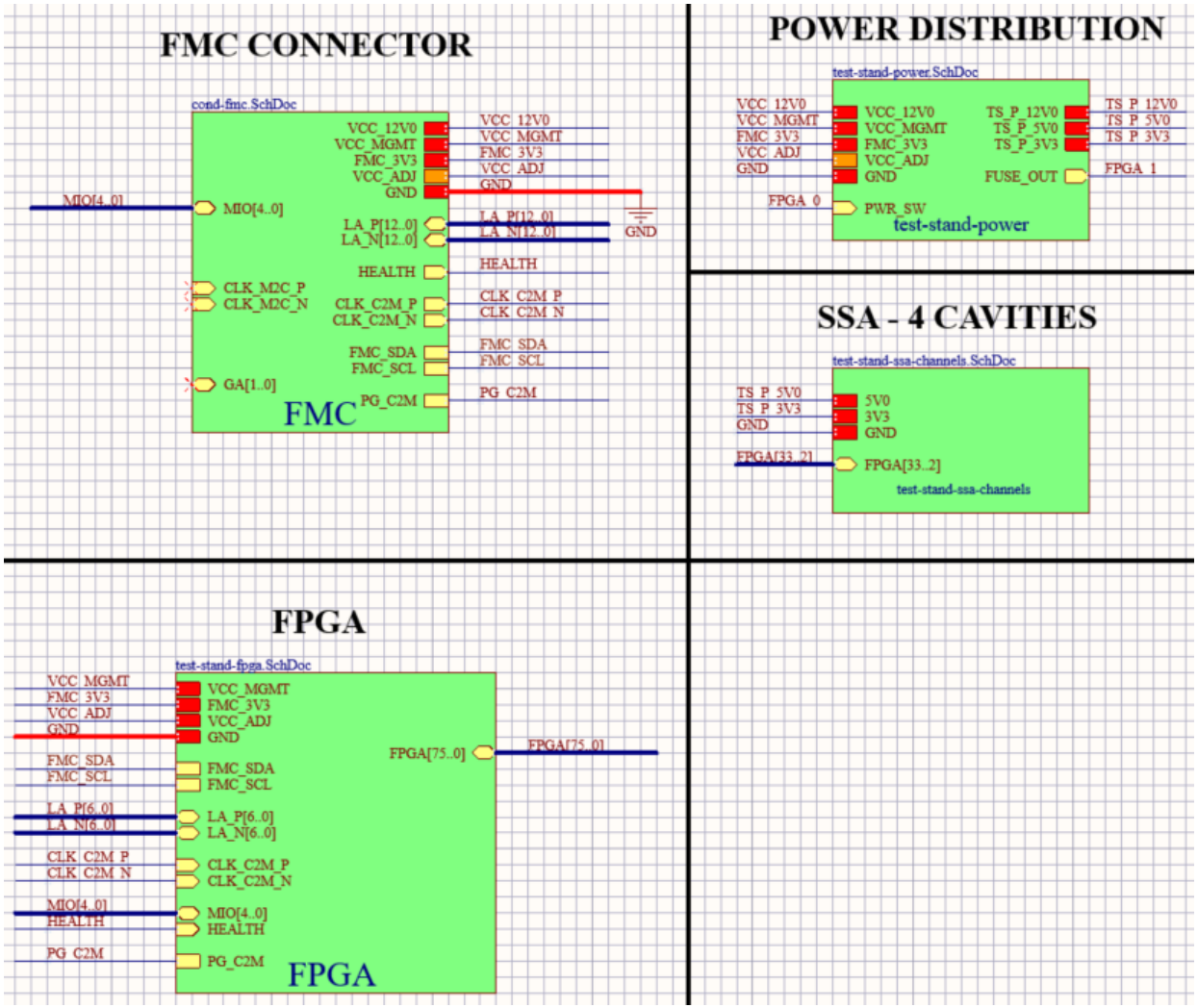
The functionality and design details

- The test stand board for RTD conditioning boards (2 channels per 1 cavity → 8 channels per RFPI system) tested for the temperature readings in the expected range (RTD sensor resistance simulated by 5 resistors). Reading done in a sequence using relays. Using 4-wire connections.



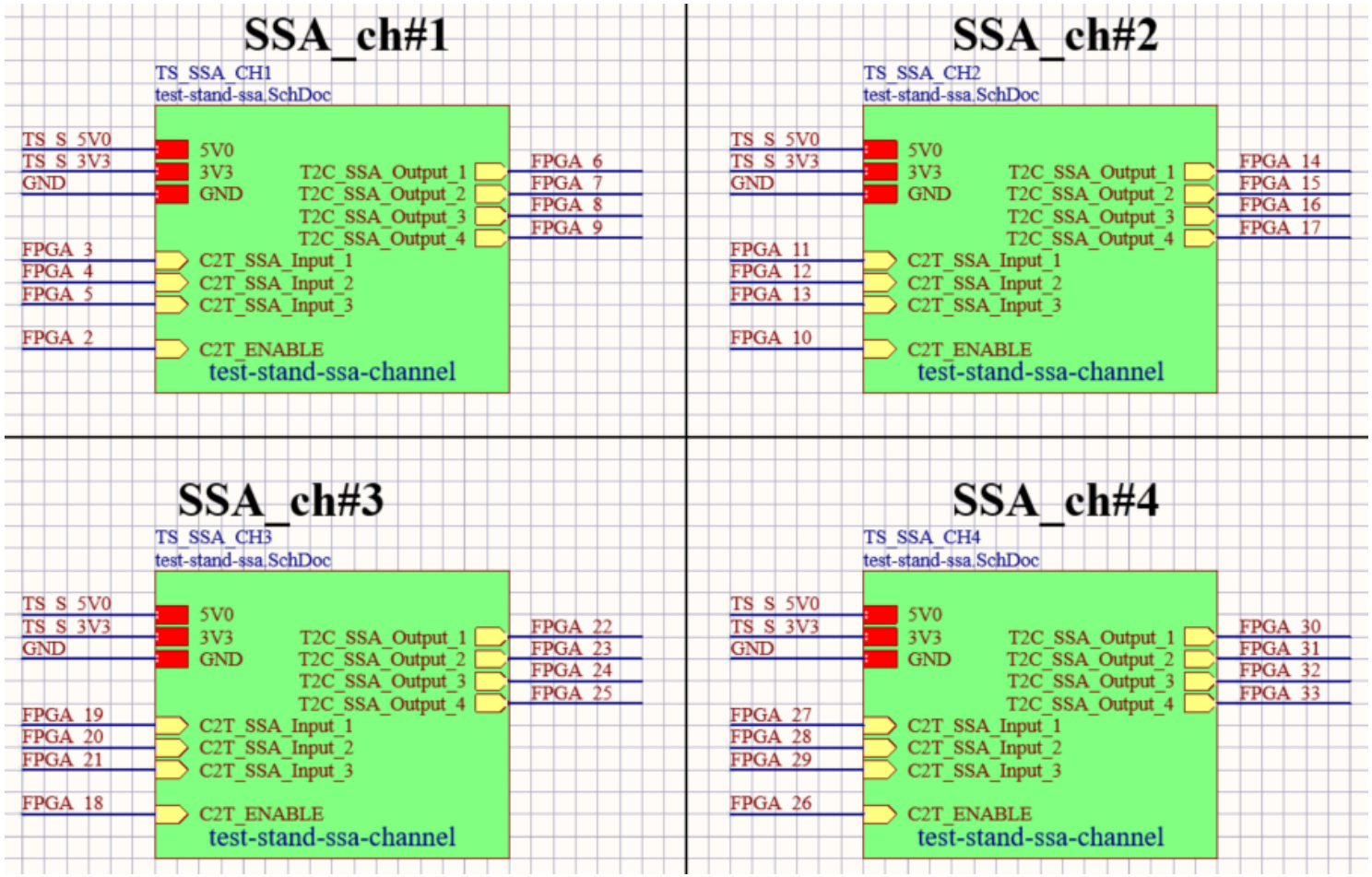
The functionality and design details

- The test stand board for SSA conditioning boards (4 output channels and 2 input channels per 1 cavity → 24 channels in total) – overall design



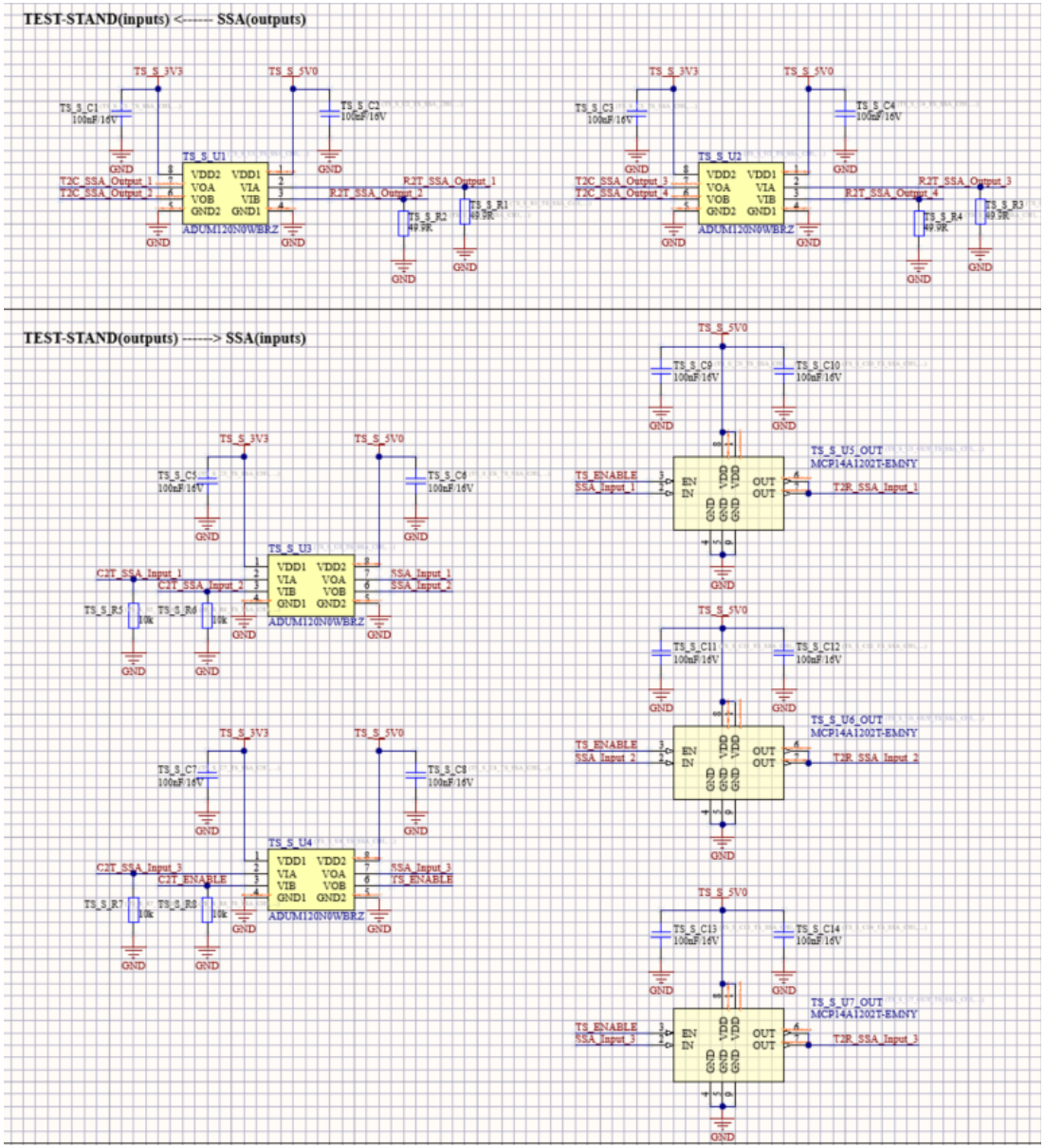
The functionality and design details

- The test stand board for SSA conditioning boards (4 output channels and 2 input channels per 1 cavity → 24 channels in total)



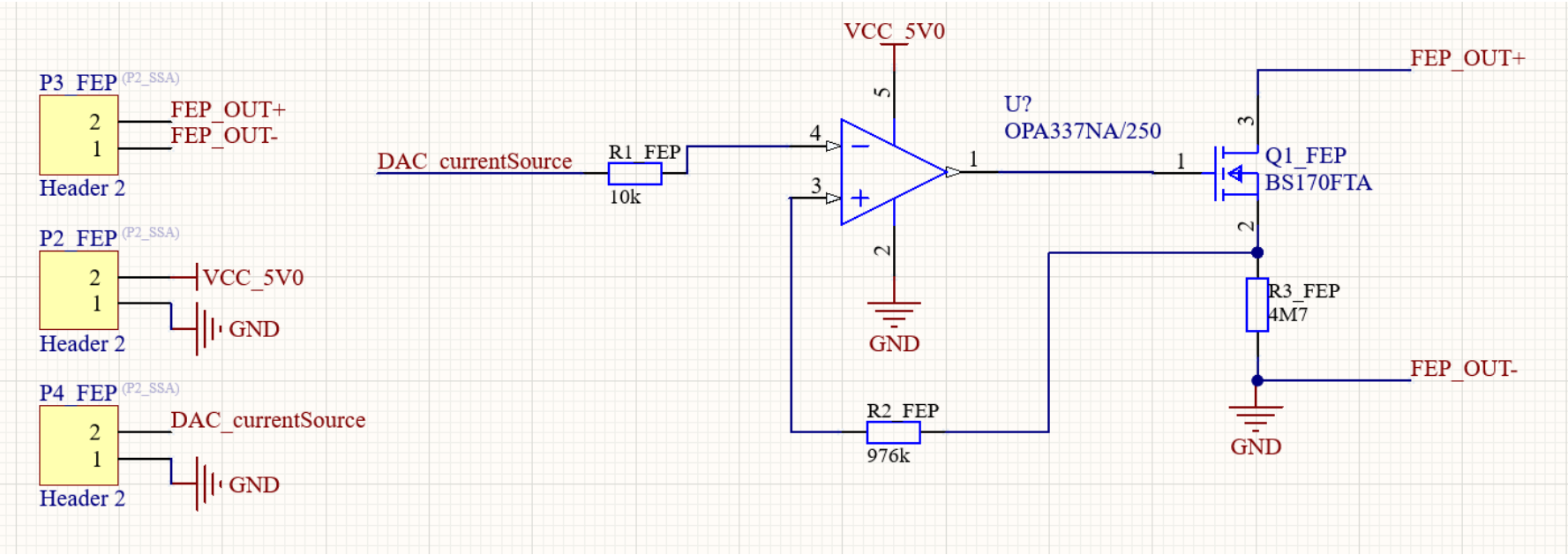
The functionality and design details

- The test stand board for SSA conditioning boards (4 output channels and 2 input channels per 1 cavity → 24 channels in total) tested for time constraints for outputs and inputs
- Test stand provides 4 output channels and 3 input channels per cavity



The functionality and design details

- The test stand board for FEP conditioning boards (1 channel per cavity → 4 channels per RFPI system) tested for:
  - current measurement by using DAC controlled current source on MOSFET



The functionality and design details

- The NIRP filters tested with the use of external RF generator (ie. RIGOL DSG830) controlled via Ethernet/USB connection and VISA protocol (ie. pyVISA library for Python scripts).

```
#SDG PyVISA LAN Arb
#Creates a 10 point stepped arbitrary waveform.
#The step starts with least significant bit (LSB) x 2, (Middle bit - L
#Most significant bit (MSB - Middle Bit)/2 x 2, and MSB x 2
#
#Sends using PyVISA LAN connection
#
#Author: JAC
#Date: 04/24/2020
#
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import visa
import time
import binascii
import random

#VISA resource of Device
device_resource = 'TCPIP0::192.168.0.19::inst0::INSTR'

#Little endian, 16-bit 2's complement

#send 10 points to the generator.
#16 bit for SDG6000X and SDG2000X
#hex(-32768)= 0X8000, hex(-16384)=0XC000, hex(0)=0X0000, hex(16384)=0X
#wave_data = [0x8000, 0x8000, 0xc000, 0xc000, 0x0000, 0x0000, 0x4000,
#
#14 bit data for SDG1000X
#hex(-8192)= 0XE000, hex(-4096)=0XF000, hex(0)=0X0000, hex(4096)=0X10
wave_data = [0xe000, 0xe000, 0xf000, 0xf000, 0x0000, 0x0000, 0x1000, 0
wave_points = []

for i in range(0,1):
    wave_points += wave_data
count=1

print(wave_points)
print(len(wave_points))

def create_wave_file():
    """create a file"""
    f = open("wave1.bin", "wb")
    for a in wave_points:
        b = hex(a)
        b = b[2:]
        len_b = len(b)
        if (0 == len_b):
            b = '0000'
```

Full scale design plans

- Adding HV test board
- Validation of all test boards
- Extended EPICS interface for automatic tests

Summary

Thank you for your attention