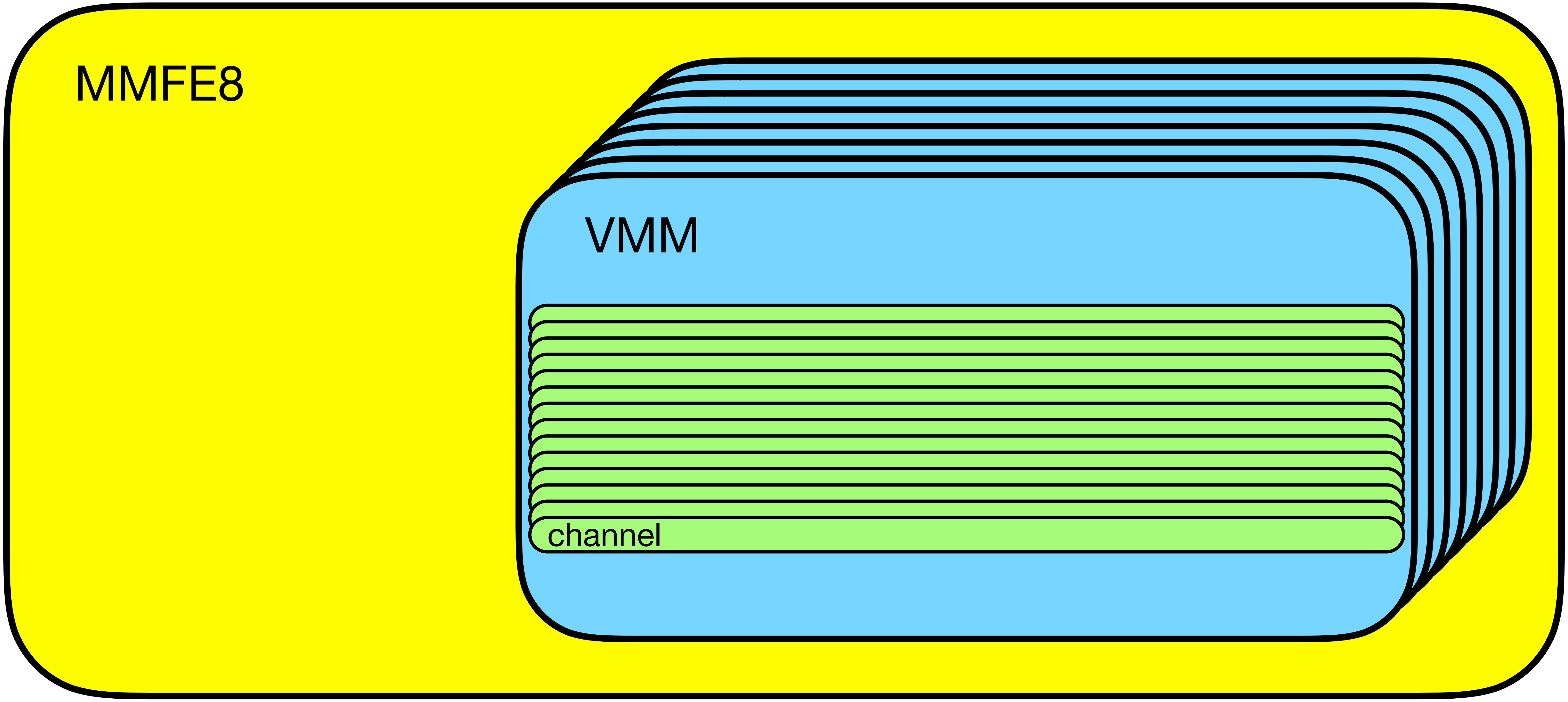
Arizona GUI

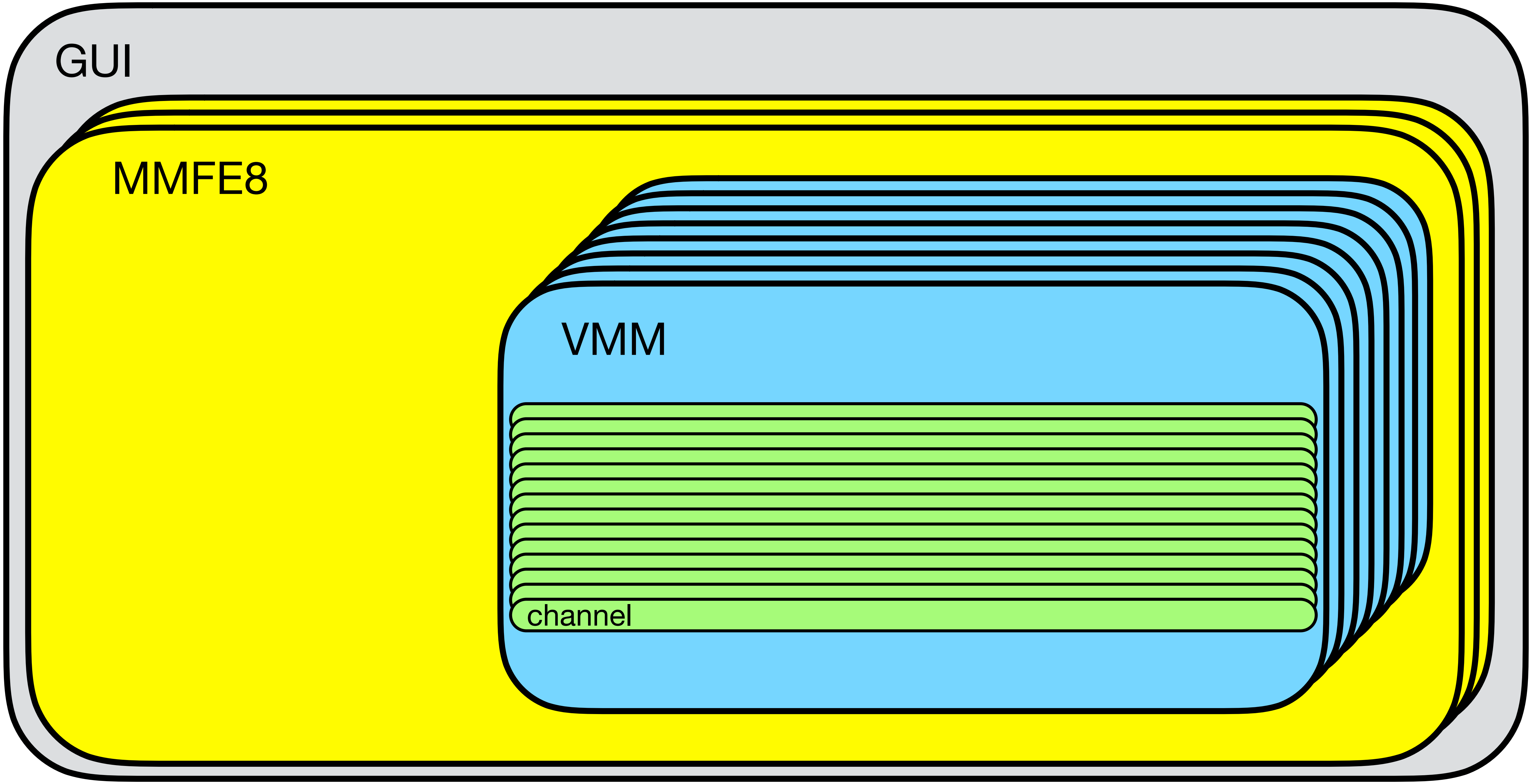
GUI = MMFE8
MMFE has 8 VMM. VMM has 64 channels.



multi-MMFE GUI

GUI = GUI

GUI has N MMFE. MMFE has 8 VMM. VMM has 64 channels.



code, how-to

https://github.com/alexandertuna/MMFE8Readout/tree/master/python/dev/tuna/

```
$ python gui.py

Loading MMFE8 GUI with 1 MMFE

Creating instance of MMFE

Setting default values
[clipped]

Goodbye from the MMFE8 GUI!
$
$
$ python gui.py 3

Loading MMFE8 GUI with 3 MMFE

Creating instance of MMFE
Creating instance of MMFE
Creating instance of MMFE

Setting default values
[clipped]
```

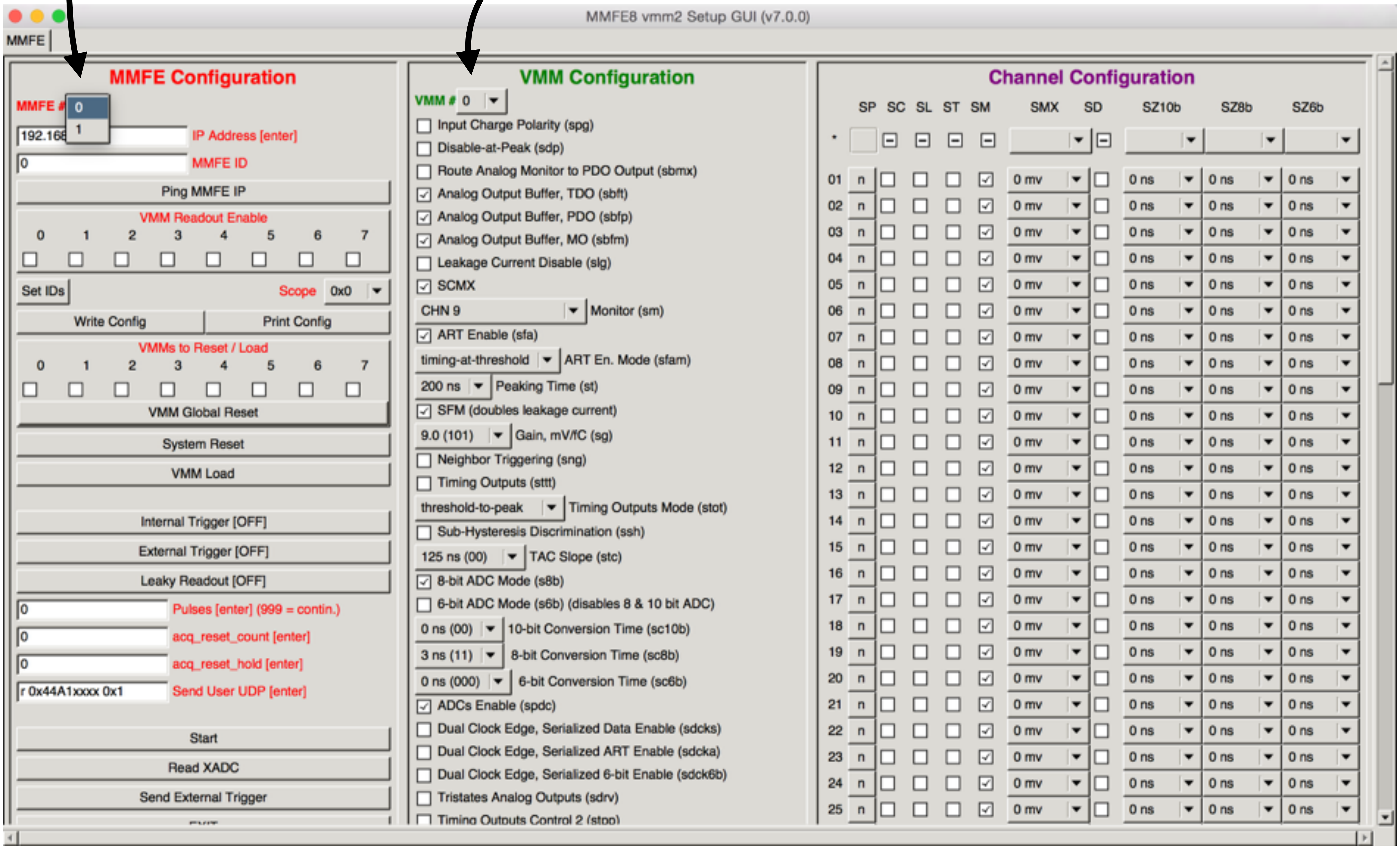
looks a lot like existing GUI

number of MMFE to connect with:
configurable from command line

works with any* version of the firmware
(same as existing GUI)

control which MMFE to talk with

control which VMM to configure



MMFE8 vmm2 Setup GUI (v7.0.0)

MMFE

MMFE Configuration

MMFE # 0 ▾ (0, 1)

192.168 — IP Address [enter]

0 — MMFE ID

Ping MMFE IP

VMM Readout Enable

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□

Set IDs — Scope 0x0 ▾

Write Config | Print Config

VMMs to Reset / Load

0	1	2	3	4	5	6	7
□	□	□	□	□	□	□	□

VMM Global Reset

System Reset

VMM Load

Internal Trigger [OFF]

External Trigger [OFF]

Leaky Readout [OFF]

0 — Pulses [enter] (999 = contin.)

0 — acq_reset_count [enter]

0 — acq_reset_hold [enter]

r 0x44A1xxxx 0x1 — Send User UDP [enter]

Start

Read XADC

Send External Trigger

VMM Configuration

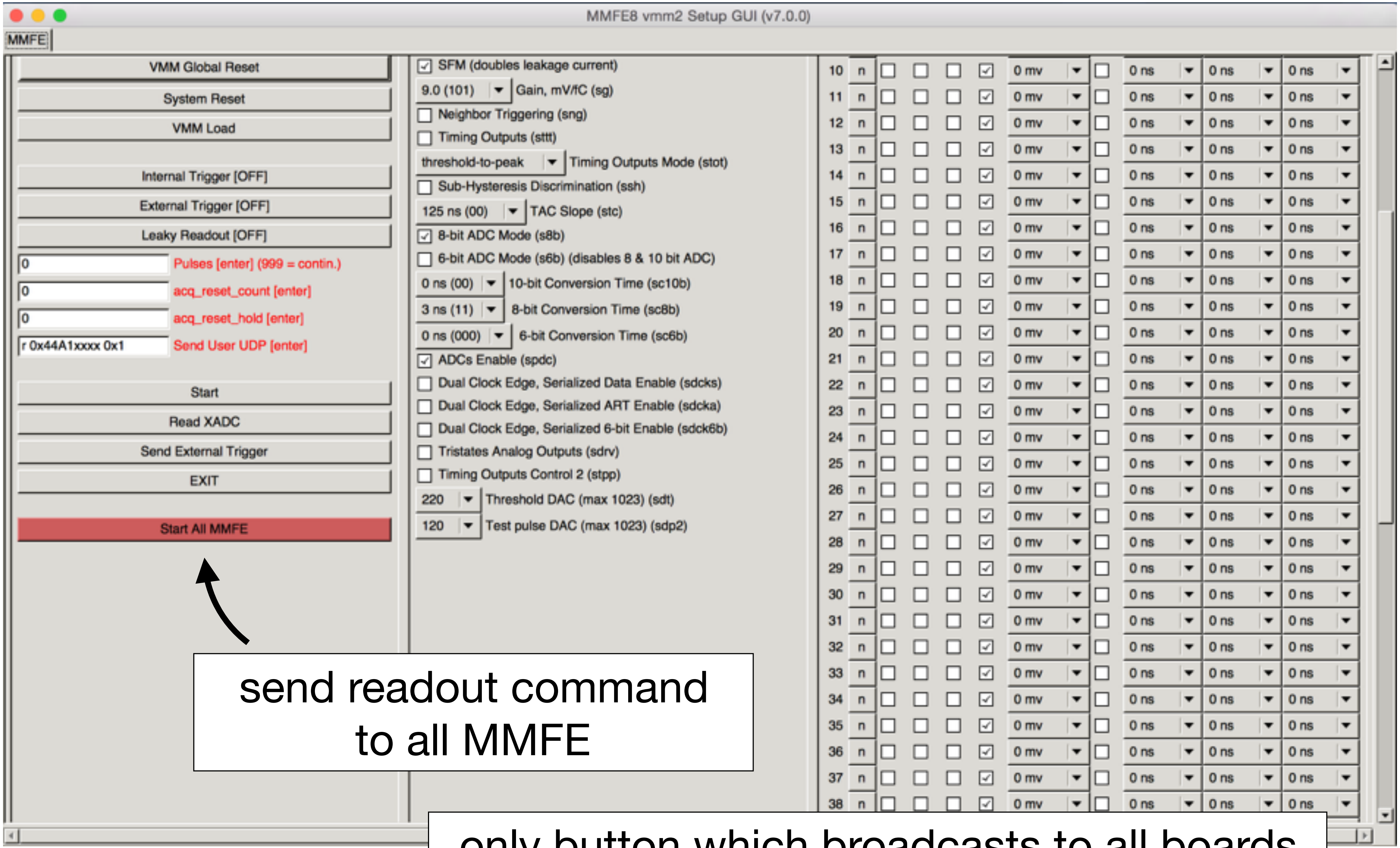
VMM # 0 ▾

- □ Input Charge Polarity (spg)
- □ Disable-at-Peak (sdp)
- □ Route Analog Monitor to PDO Output (sbmx)
- ☑ Analog Output Buffer, TDO (sbft)
- ☑ Analog Output Buffer, PDO (sbfp)
- ☑ Analog Output Buffer, MO (sbfm)
- □ Leakage Current Disable (slg)
- ☑ SCMX
- CHN 9 ▾ Monitor (sm)
- ☑ ART Enable (sfa)
- timing-at-threshold ▾ ART En. Mode (sfam)
- 200 ns ▾ Peaking Time (st)
- ☑ SFM (doubles leakage current)
- 9.0 (101) ▾ Gain, mV/fC (sg)
- □ Neighbor Triggering (sng)
- □ Timing Outputs (sttt)
- threshold-to-peak ▾ Timing Outputs Mode (stot)
- □ Sub-Hysteresis Discrimination (ssh)
- 125 ns (00) ▾ TAC Slope (stc)
- ☑ 8-bit ADC Mode (s8b)
- □ 6-bit ADC Mode (s6b) (disables 8 & 10 bit ADC)
- 0 ns (00) ▾ 10-bit Conversion Time (sc10b)
- 3 ns (11) ▾ 8-bit Conversion Time (sc8b)
- 0 ns (000) ▾ 6-bit Conversion Time (sc6b)
- ☑ ADCs Enable (spdc)
- □ Dual Clock Edge, Serialized Data Enable (sdcks)
- □ Dual Clock Edge, Serialized ART Enable (sdcka)
- □ Dual Clock Edge, Serialized 6-bit Enable (sdck6b)
- □ Tristates Analog Outputs (sdrv)
- □ Timing Outputs Control 2 (stpp)

Channel Configuration

	SP	SC	SL	ST	SM	SMX	SD	SZ10b	SZ8b	SZ6b
*		⊟	⊟	⊟	⊟	▾	⊟	▾	▾	▾
01	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
02	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
03	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
04	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
05	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
06	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
07	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
08	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
09	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
10	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
11	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
12	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
13	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
14	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
15	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
16	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
17	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
18	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
19	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
20	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
21	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
22	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
23	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
24	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾
25	n	□	□	□	☑	0 mv ▾	□	0 ns ▾	0 ns ▾	0 ns ▾

after setting up MMFEs independently (including test pulses),
scroll down …



MMFE8 vmm2 Setup GUI (v7.0.0)

MMFE

0	Pulses [enter] (999 = contin.)
0	acq_reset_count [enter]
0	acq_reset_hold [enter]
r 0x44A1xxxx 0x1	Send User UDP [enter]

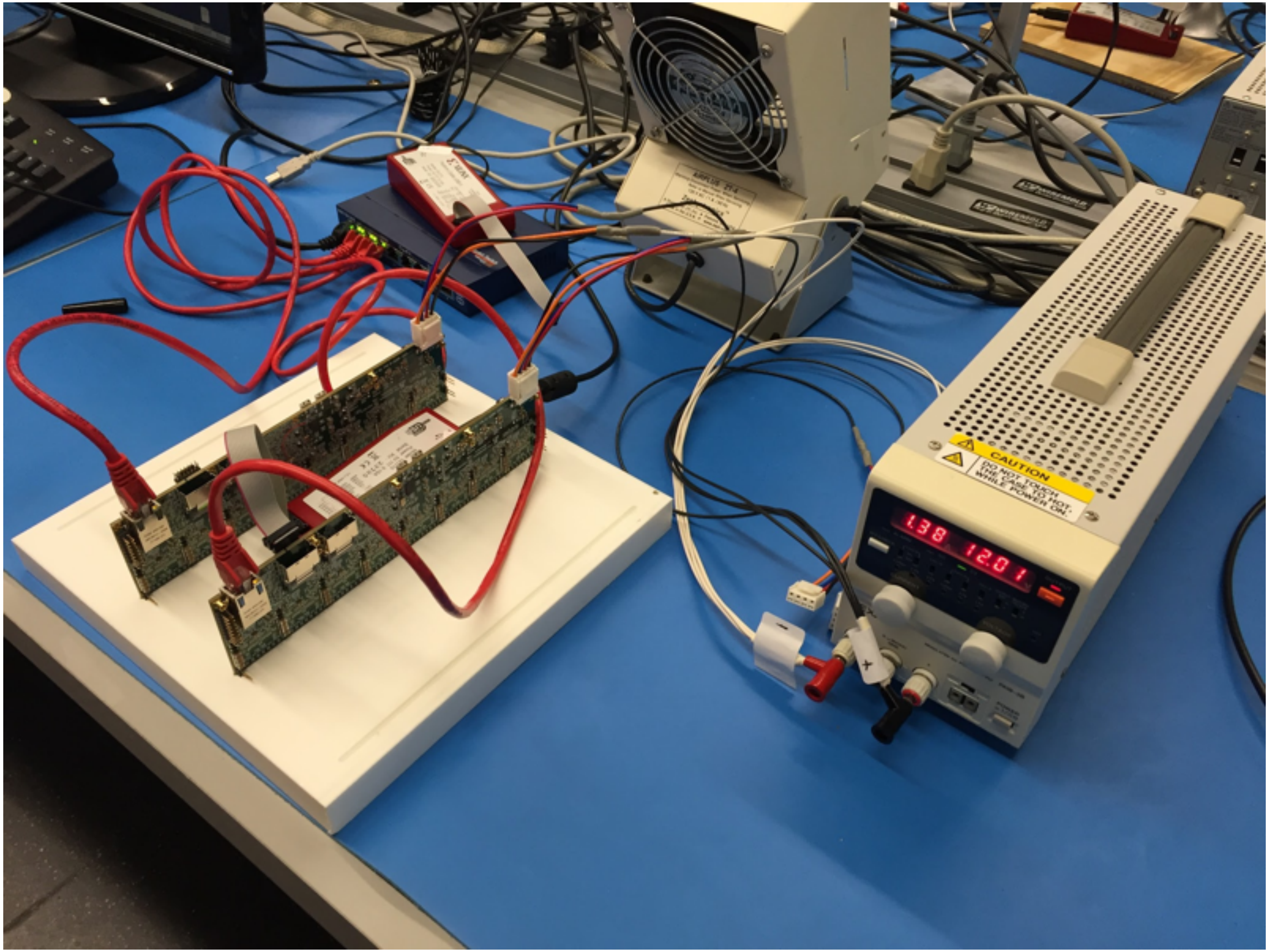
☑ SFM (doubles leakage current)
9.0 (101) Gain, mV/fC (sg)
□ Neighbor Triggering (sng)
□ Timing Outputs (sttt)
threshold-to-peak Timing Outputs Mode (stot)
□ Sub-Hysteresis Discrimination (ssh)
125 ns (00) TAC Slope (stc)
☑ 8-bit ADC Mode (s8b)
□ 6-bit ADC Mode (s6b) (disables 8 & 10 bit ADC)
0 ns (00) 10-bit Conversion Time (sc10b)
3 ns (11) 8-bit Conversion Time (sc8b)
0 ns (000) 6-bit Conversion Time (sc6b)
☑ ADCs Enable (spdc)
□ Dual Clock Edge, Serialized Data Enable (sdcks)
□ Dual Clock Edge, Serialized ART Enable (sdcka)
□ Dual Clock Edge, Serialized 6-bit Enable (sdck6b)
□ Tristates Analog Outputs (sdrv)
□ Timing Outputs Control 2 (stpp)
220 Threshold DAC (max 1023) (sdt)
120 Test pulse DAC (max 1023) (sdp2)

10	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
11	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
12	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
13	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
14	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
15	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
16	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
17	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
18	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
19	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
20	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
21	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
22	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
23	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
24	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
25	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
26	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
27	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
28	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
29	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
30	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
31	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
32	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
33	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
34	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
35	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
36	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
37	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns
38	n	□	□	□	☑	0 mv	□	0 ns	0 ns	0 ns

send readout command
to all MMFE

only button which broadcasts to all boards

testing



testing

left board

right board

```
Sending 'w 0x44A100FC 0x4 \x00\n' to 192.168.0.101 :: 50001
Receive 'OK\n'
Sending 'r 0x44A10014 1 \x00\n' to 192.168.0.101 :: 50001
Receive 'R 0x44a10014 0x14 \n'
Closing socket

FIFOCNT  20
Sending 'k 0x44A10010 10 \x00\n' to 192.168.0.101 :: 50001
Receive 'K 0x44a10010 0xe07033 0xbc30a65d 0xe07037 0xbc30a65d 0x765037 0xbc36b19d 0x665433 0xbc36b19f 0x5a5433 0xbc3cd5df \n'
Closing socket

word0 = 0x7e7037 word1 = 0xbc3cd5df addr = 14 amp = 624 time = 31 bcid = 1685 vmm = 0 mmfe = 60
word0 = 0x665033 word1 = 0xbc42fa1e addr = 13 amp = 592 time = 25 bcid = 3092 vmm = 0 mmfe = 66
word0 = 0x6e6037 word1 = 0xbc42fa1f addr = 14 amp = 608 time = 27 bcid = 3093 vmm = 0 mmfe = 66
[clipped]

Sending 'w 0x44A100FC 0x4 \x00\n' to 192.168.0.102 :: 50001
Receive 'OK\n'
Sending 'r 0x44A10014 1 \x00\n' to 192.168.0.102 :: 50001
Receive 'R 0x44a10014 0x14 \n'
Closing socket

FIFOCNT  20
Sending 'k 0x44A10010 10 \x00\n' to 192.168.0.102 :: 50001
Receive 'K 0x44a10010 0x4e0703f 0xb9b9babf 0x4e0703b 0xb9b9babf 0x479203b 0xb9bfcb7d 0x491503f 0xb9bfcb7f 0x46d203b 0xb9c5ec3f \n'
Closing socket

word0 = 0x471203b word1 = 0xb9cc03fe addr = 15 amp = 288 time = 28 bcid = 683 vmm = 1 mmfe = 204
word0 = 0x481503f word1 = 0xb9cc03fe addr = 16 amp = 336 time = 32 bcid = 683 vmm = 1 mmfe = 204
```

(need to ping board when switching, unclear why. now built in.)

bonus

in practice

GUI class

```
self.MMFEs = []
for i in xrange(nmmfes):
    self.MMFEs.append(MMFE())
```

MMFE class

```
self.VMMs = []
for i in range(nvmms):
    self.VMMs.append(VMM())
```

VMM class

```
self.channels = []
for ch in xrange(64):
    self.channels.append(Channel(ch))
```

what info is stored in GUI?

GUI class

all GUI attributes: buttons, windows, et al

MMFE class

`mmfe.udp*`	: UDP communication
`mmfe.vmm_cfg_sel`	: VMM configurations
`mmfe.readout_runlength`	: readout, triggering
`mmfe.control`	: control stuff

VMM class

`vmm.globalreg:`	VMM-level configuration
`vmm.reg` :	channels configuration for that VMM

channel class

`channel.value:` channel-level options for that channel

consolidating code

```python
def start(self, widget):
    self.control[2] = 1
    self.write_control()

    self.daq_readOut()
    time.sleep(1)

    self.control[2] = 0
    self.write_control()

def load_IDs(self):
    self.write_vmm_cfg_sel()
    self.write_readout_runlength()

def write_control(self):
    message = "w 0x44A100FC 0x{0:X}".format(convert_to_32bit(self.control))
    self.udp.udp_client(message, self.UDP_IP, self.UDP_PORT)

def write_readout_runlength(self):
    message = "w 0x44A100F4 0x{0:X}".format(convert_to_32bit(self.readout_runlength))
    self.udp.udp_client(message, self.UDP_IP, self.UDP_PORT)

def write_vmm_cfg_sel(self):
    message = "w 0x44A100EC 0x{0:X}".format(convert_to_32bit(self.vmm_cfg_sel))
    self.udp.udp_client(message, self.UDP_IP, self.UDP_PORT)
```

previous code typically
copy-pasted

consolidating code

```
wc -l gui.py mmfe.py vmm.py channel.py helpers.py
     975 gui.py
     353 mmfe.py
      66 vmm.py
      27 channel.py
       9 helpers.py
    1430 total
```

```
wc -l mmfe8_v7_bhx.py mmfe8_vmm.py mmfe8_chan.py mmfe8_userRegs.py
    1395 mmfe8_v7_bhx.py
    1360 mmfe8_vmm.py
     267 mmfe8_chan.py
     528 mmfe8_userRegs.py
    3550 total
```

additional steering

```
nmmfes    = 1 if len(sys.argv)==1 else int(sys.argv[1])
nvmms     = 8
nchannels = 64
```

```
def convert_to_int(list_of_bits):
    this = "0b"
    for bit in list_of_bits:
        this += str(bit)
    return int(this, base=2)

def convert_to_32bit(list_of_bits):
    return sum([int(list_of_bits[bit])*pow(2, bit) for bit in xrange(32)])
```

additional steering

VMM registers

```
class registers:
    SPG     = 16 # input charge polarity
    SDP     = 17 # disable at peak
    SBMX    = 18 # route analog monitor to pdo output
    SBFT    = 19 # analog output buffers enable tdo
    SBFP    = 20 # analog output buffers enable pdo
    SBFM    = 21 # analog output buffers enable mo
    SLG     = 22 # leakage current disable
    SM      = 23 # monitor multiplexing
    SCMX    = 29 # monitor multiplexing enable
    SFA     = 30 # ART enable
    SFAM    = 31 # ART mode
    ST      = 32 # peaking time
    SFM     = 34 # UNKNOWN
    SG      = 35 # gain
    SNG     = 38 # neighbor triggering enable
    STOT    = 39 # timing outputs control
    STTT    = 40 # timing outputs enable
    SSH     = 41 # sub-hysteresis discrimination enable
    STC     = 42 # TAC slope adjustment
    SDT     = 44 # course threshold DAC
    SDP2    = 54 # test pulse DAC
    SC10b   = 65 # 10-bit ADC conversion time
    SC8b    = 67 # 8-bit ADC conversion time
    SC6b    = 70 # 6-bit ADC conversion time
    S8b     = 71 # 8-bit ADC conversion mode
    S6b     = 72 # 6-bit ADC conversion enable
    SPDC    = 73 # ADCs enable
    SDCKS   = 74 # dual clock edge serialized data enable
    SDCKA   = 75 # dual clock edge serialized ART enable
    SDCK6b  = 76 # dual clock edge serialized 6-bit enable
    SDRV    = 77 # tristates analog outputs with token, used in analog mode
    STPP    = 78 # timing outputs control 2
```

channel registers

```
class index:
    SP    =  0 # input charge polarity
    SC    =  1 # large sensor capacitance mode
    SL    =  2 # leakage current disable
    ST    =  3 # test capacitor enable
    SM    =  4 # mask enable
    SD    =  5 # threshold DAC
    SMX   =  9 # channel monitor mode
    SZ10b = 10 # 10-bit ADC
    SZ8b  = 15 #  8-bit ADC
    SZ6b  = 19 #  6-bit ADC

    bits_SD     = 4
    bits_SZ10b  = 5
    bits_SZ8b   = 4
    bits_SZ6b   = 3
```