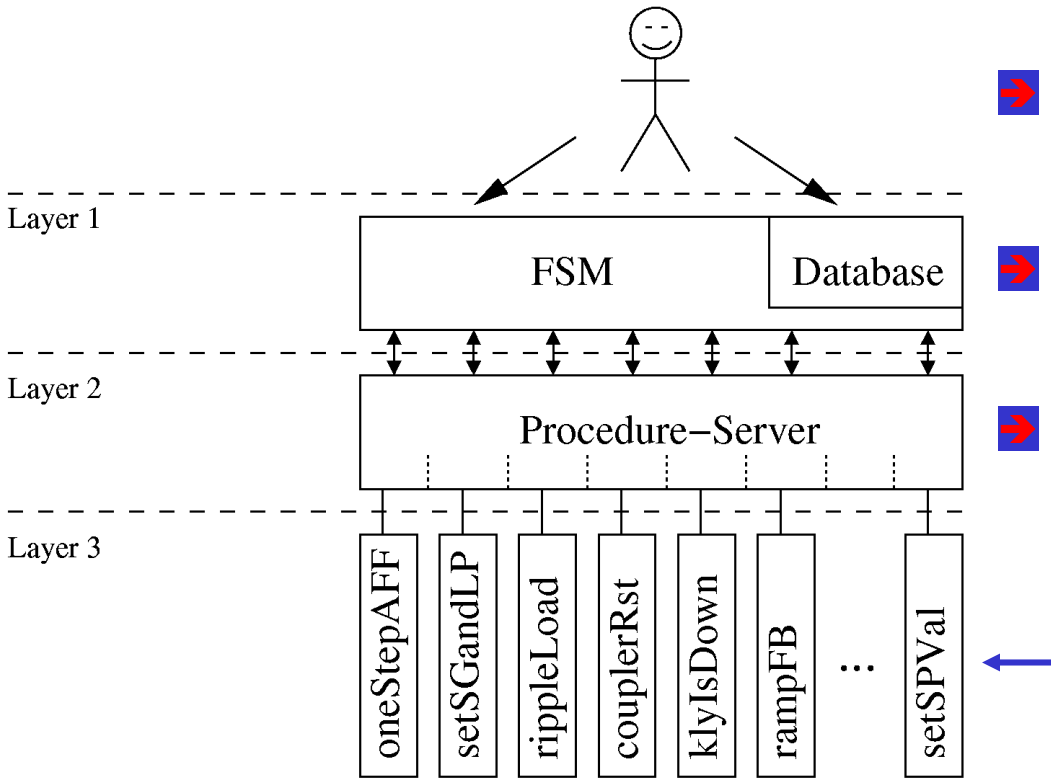
LLRF Automation and Adaptive Feedforward

alexander.brandt@desy.de

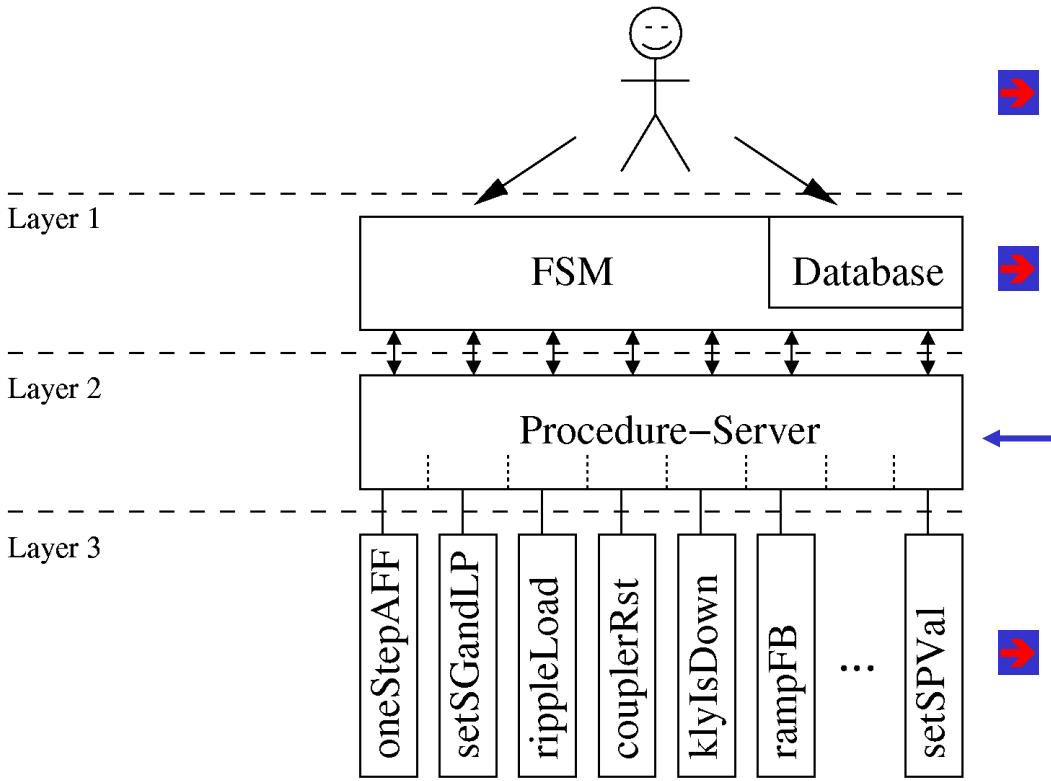
FSM-Architecture



Procedures

- Compiled scripts of any programming language (and any Matlab-version)
- "Fire and forget":
  - –Invoke - Run - Return
  - –"Stateless Procedures"
- E.g.:
  - –**Adaptive Feedforward**
  - –Loop-Phase
- Web-Documentation
- Algorithms are identical for all RF-stations and read a config file

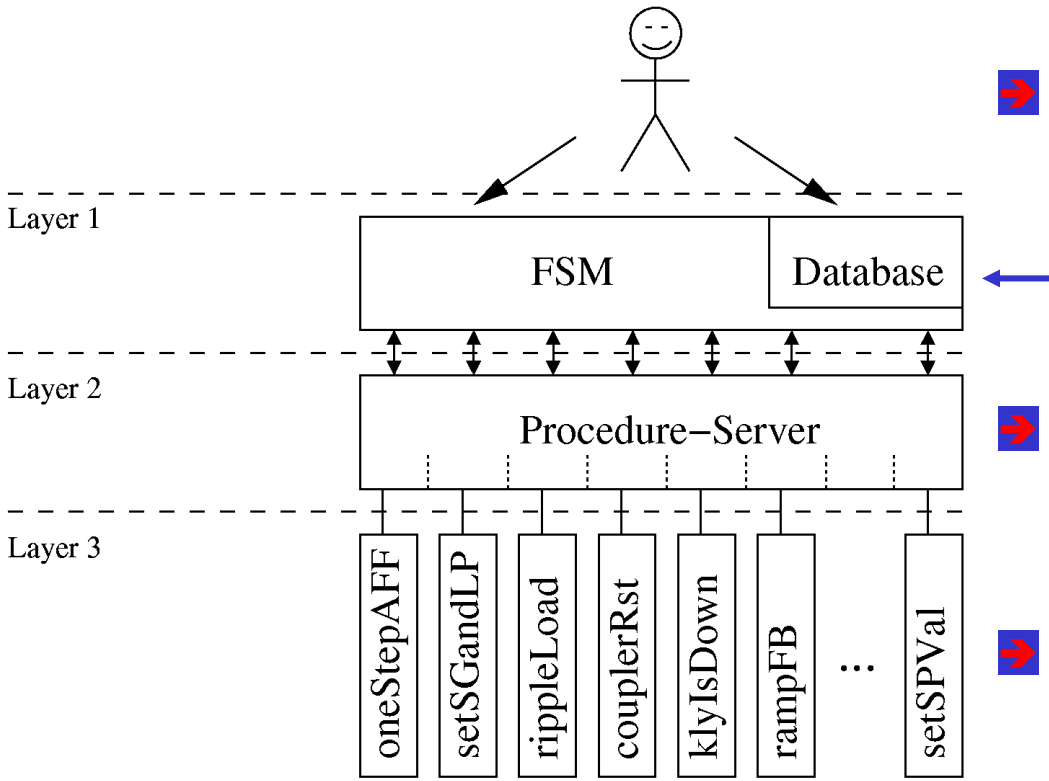
FSM-Architecture



Procedure-Server

- Doocs-interface for configuration and access
- History-Display
- Email-Notification
- Timeout-feature (free configurable)
- Multithreaded

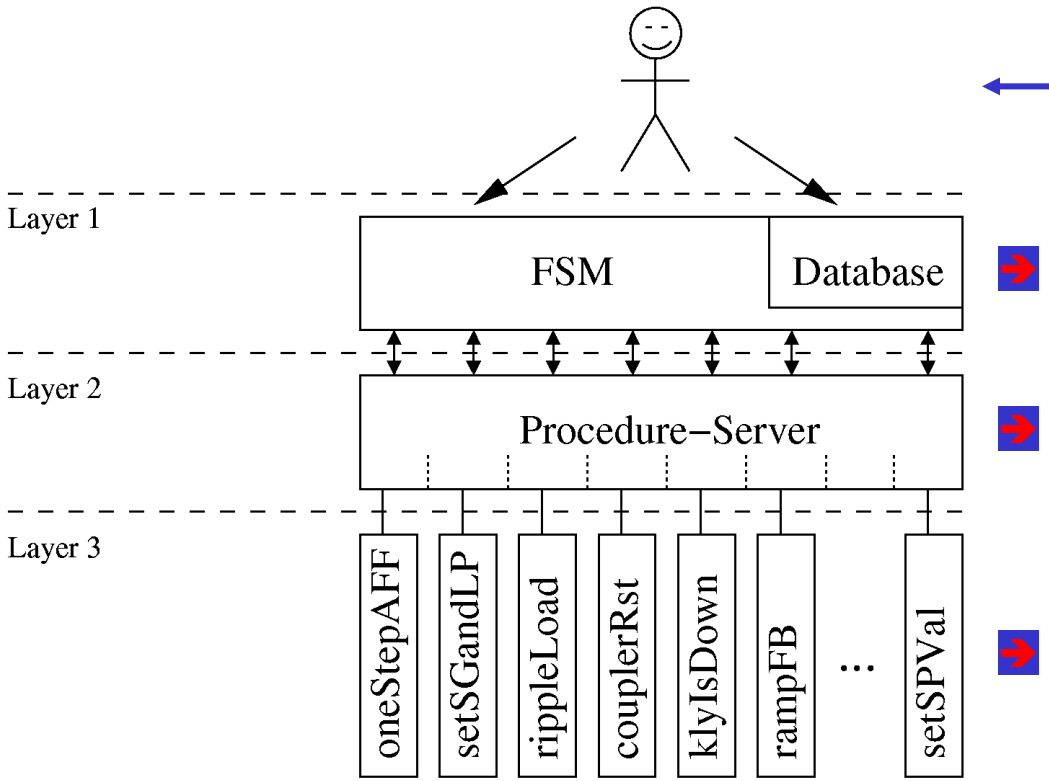
FSM-Architecture



State-Machine

- Moore-FSM:
  - –"Do something on enter"
  - –Here: trigger a procedure
- Flexible design
  - –On-line reconfigurable
  - –Off-line "restructurable"
- Current design: model the expert-operator
- "Database" stores all settings, calibration-tables, parameters for the procedures, …

FSM-Architecture

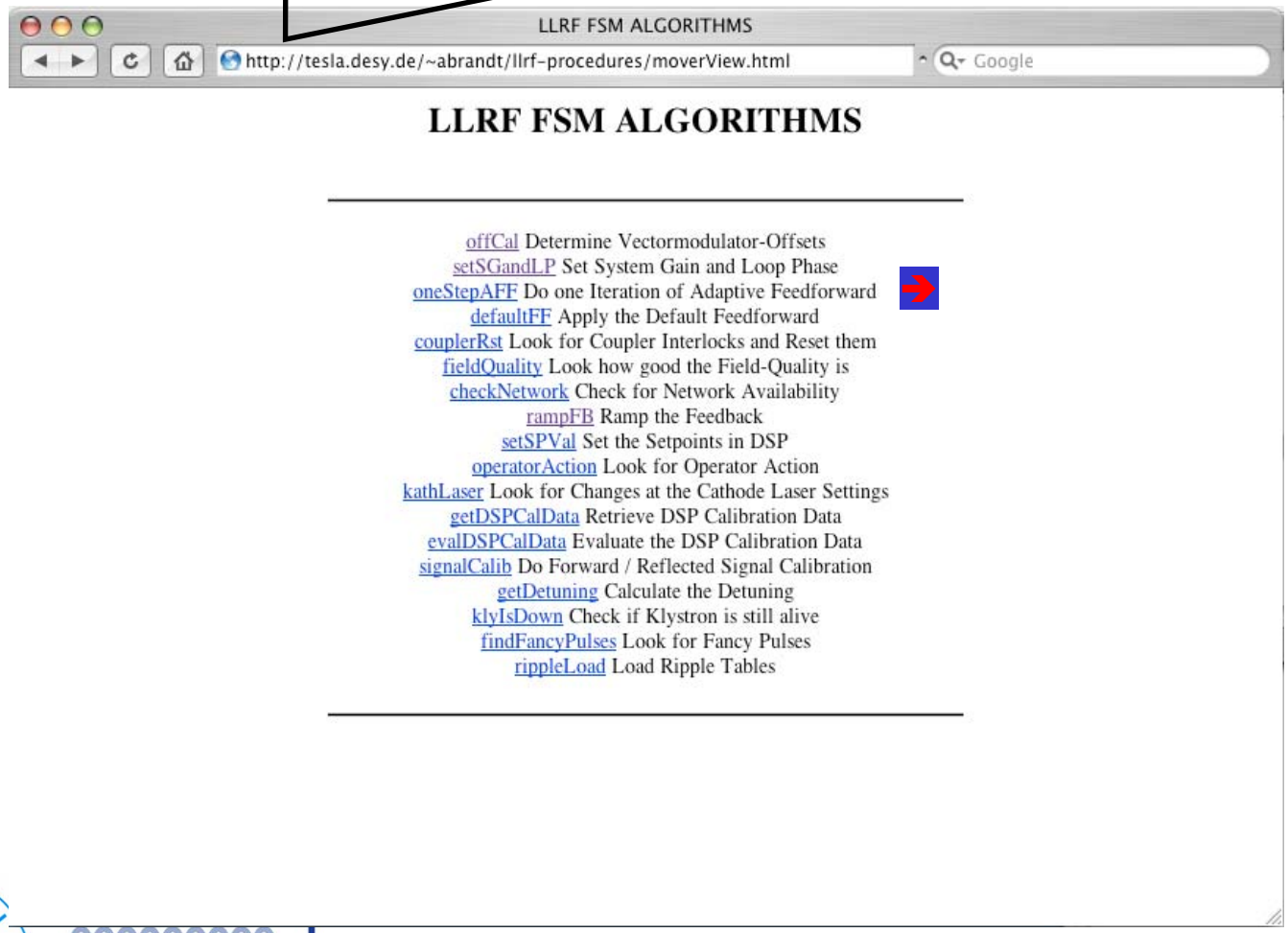


User-Interface

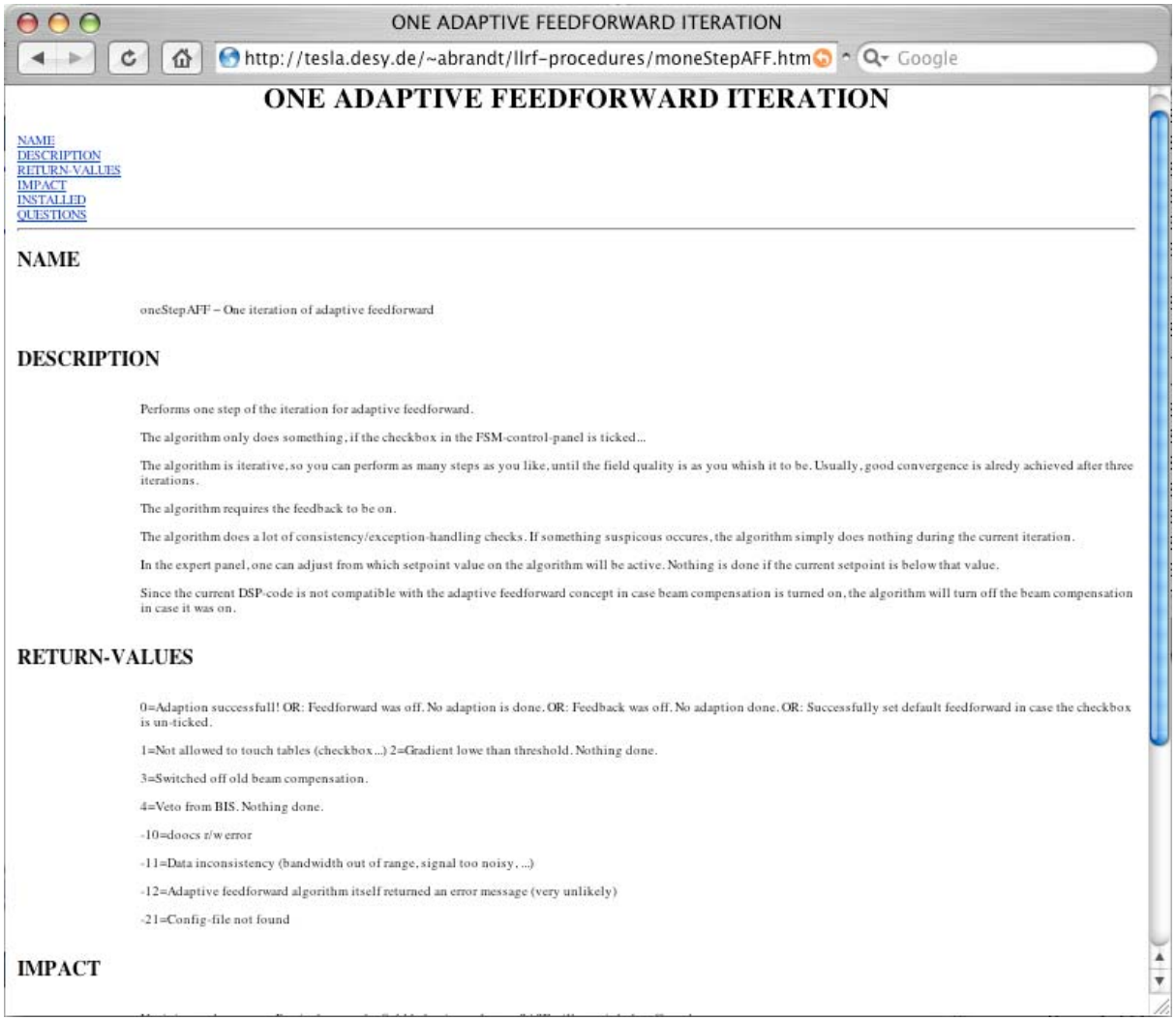
- Simple view for the operator
- Extended view for the expert

Procedure-Documentation

http://tesla.desy.de/~abrandt/llrf-procedures/moverView.html



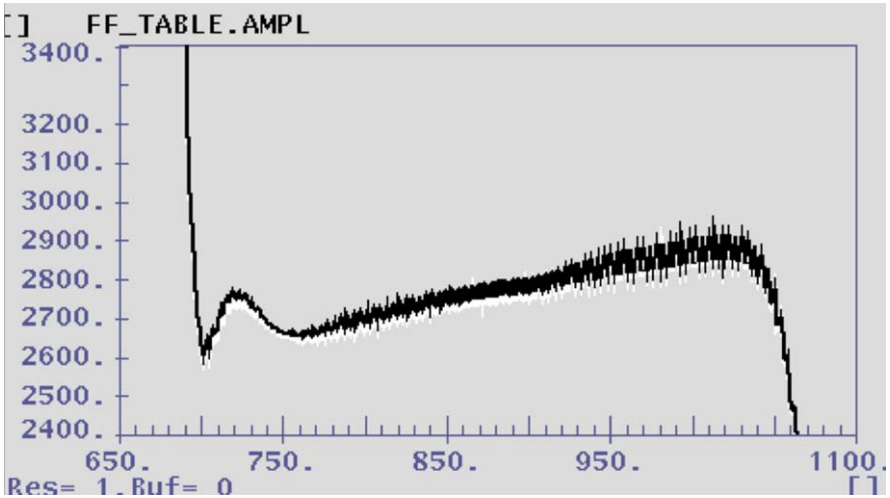
“oneStepAFF” Web-Documentation



Adaptive Feedforward

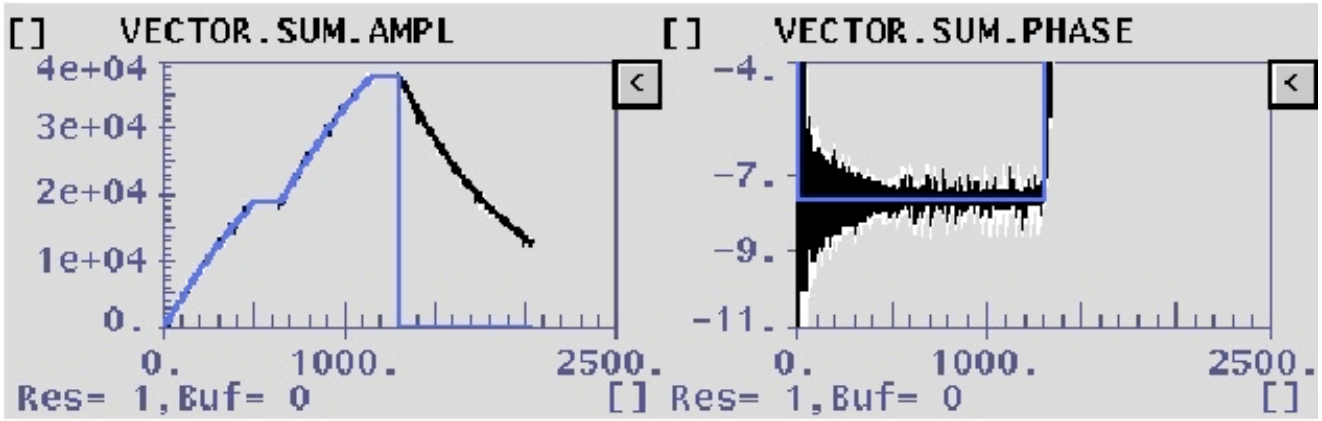
QuickTime™ and a
Cinepak decompressor
are needed to see this picture.

Adaptive Feedforward

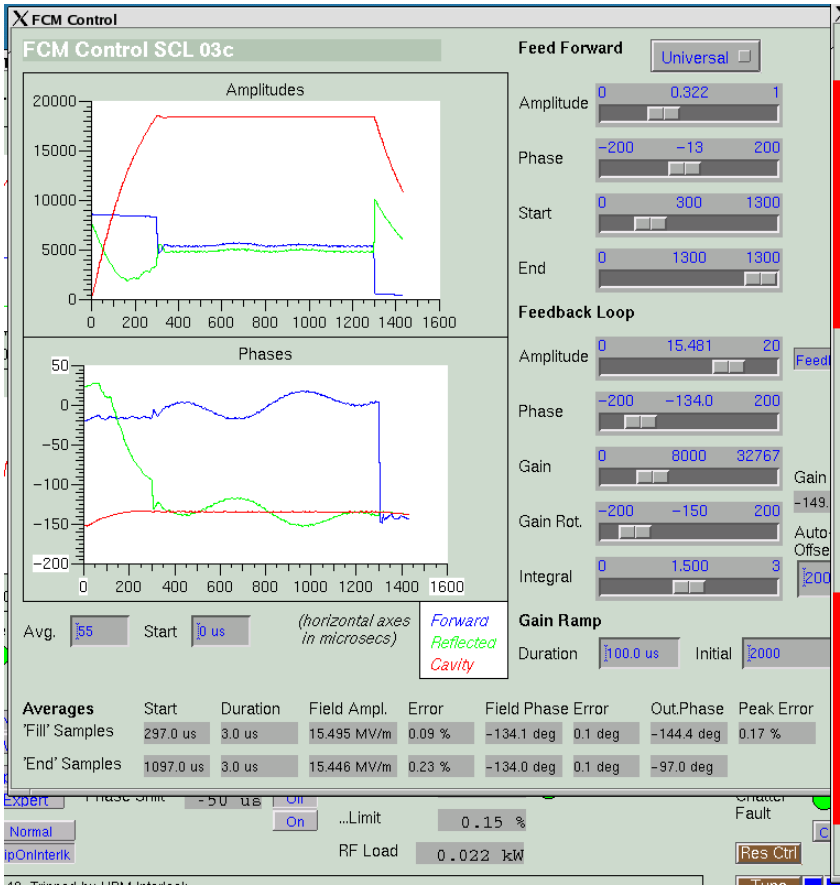


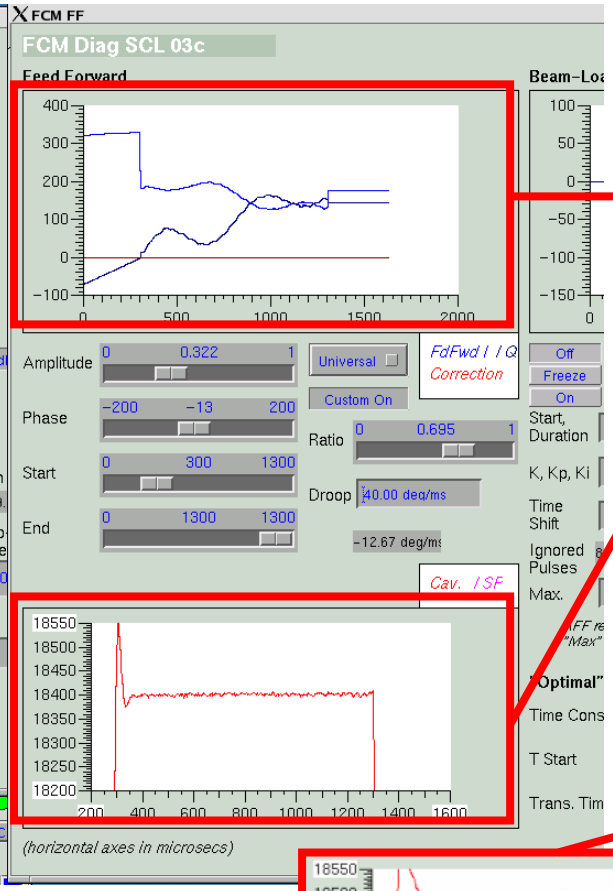
Adaptive FF w/ beam load (ACC2/3, 30us, ~1nC)
Remember, this is just the FF contribution!
E-Log 10/3/2006, 14:15

Fancy pulses w/ adaptive FF
E-Log 25/3/2006, 8:58



Adaptive Feedforward

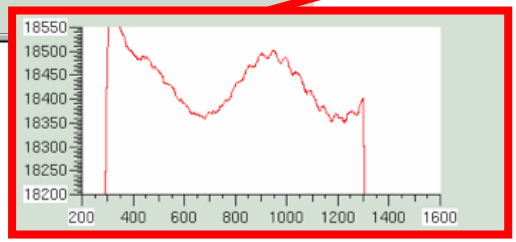




Feedforward-Table

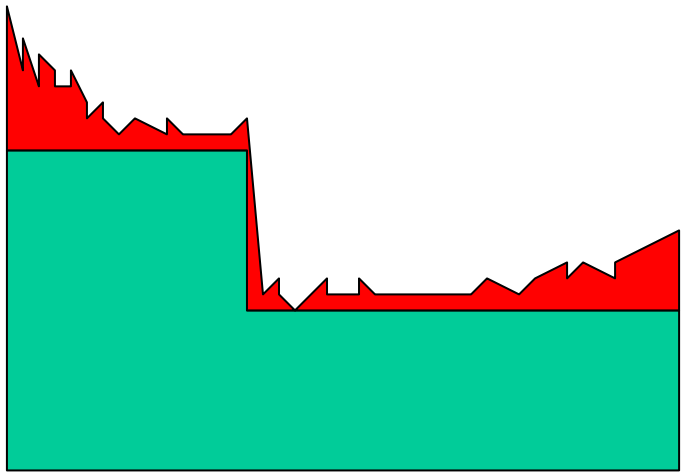
Flattop w/ Adaptive FF (3 Iterations)

Flattop w/o Adaptive FF



SNS LLRF Control Panel

Adaptive Feedforward



Feedback (FB)

Feedforward (FF)

1st idea: $FF_{new} = FB_{last} + FF_{last}$ ***…is instable :(***

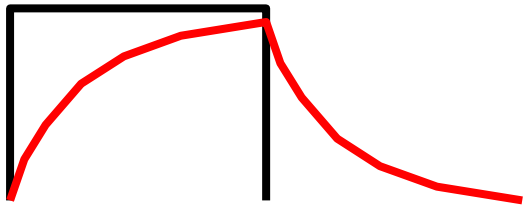
2nd idea: $FF_{new} = LP(FB_{last}) + FF_{last}$ ***…is even worse :(***

low-pass

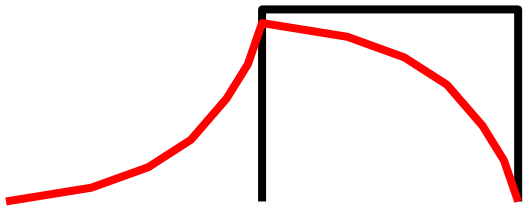
Adaptive Feedforward

3rd idea: Instead of a low-pass use a "time-reversed low-pass":

Lowpass:



Time-reversed lowpass:



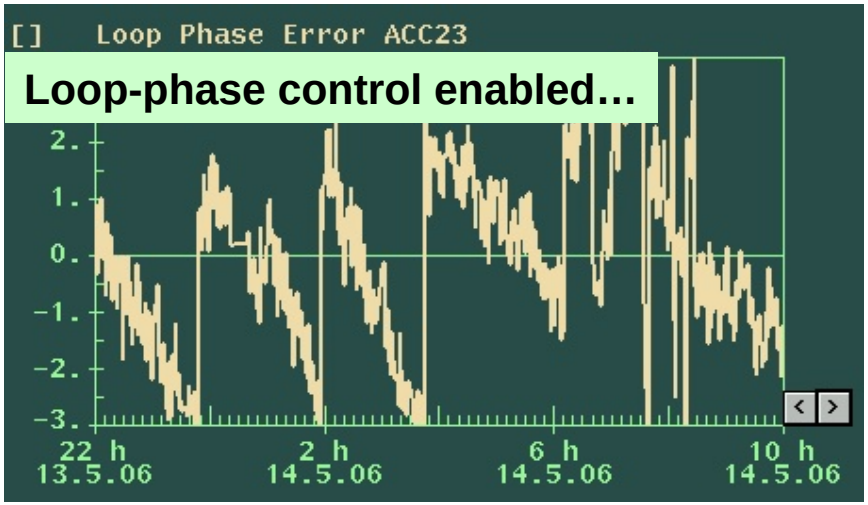
$$FF_{new} = TRLP(FB_{last}) + FF_{last}$$ ***…is surprisingly stable :)***

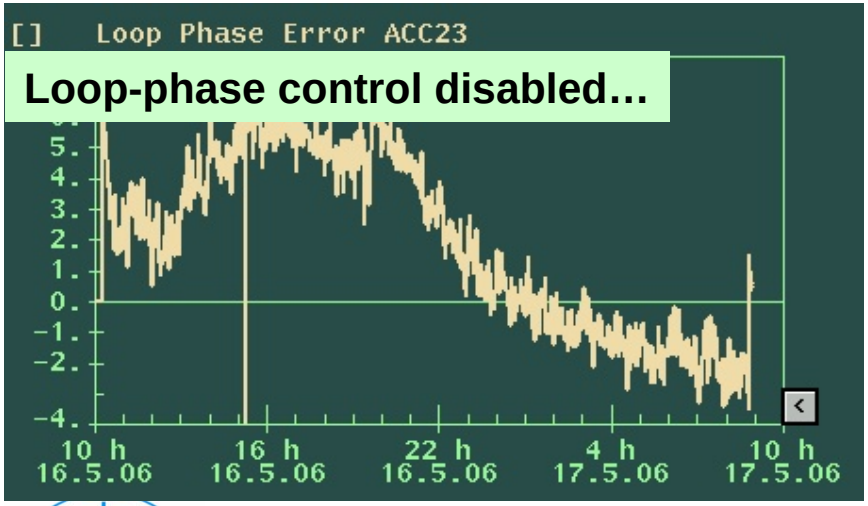
time-reversed low-pass

Adaptive Feedforward

- **Currently used: time-reversed lowpass at 2.5kHz (just 3 lines Matlab code!)**
- **Good results after 2-3 iterations**
- **However: exception handling most important!**

Loop Phase Correction





Loop Phase / System Gain Algorithm Configuration

Minimum system-gain:	+ 0.07
Maximum system-gain:	+ 0.13
Algorithm active only for SP higher than [MV/m]	+ 35.00
Enable for short pulses	☒

Config File

/home/abrandt/development/doocs/source/server/ttf/fsm/llrf_fsm_applications/confi

File Edit Text Cell Tools Debug Desktop Window Help

Stack: Base

```
203   % Adress definitions, some constants
204 -  addr_vsi = 'TTF2.RF/LLRF.DSP/ACC1/VECTOR.SUM.IPARAM';
205 -  addr_vsq = 'TTF2.RF/LLRF.DSP/ACC1/VECTOR.SUM.QPARAM';
206 -  addr_vsa = 'TTF2.RF/LLRF.DSP/ACC1/VECTOR.SUM.AMPL';
207 -  addr_vsm = 'TTF2.RF/LLRF.DSP/ACC1/VECTOR.SUM.MIX';
208 -  addr_dri = 'TTF2.RF/LLRF.DSP/ACC1/TOTAL.OUT.DAC.IPARAM';
209 -  addr_drq = 'TTF2.RF/LLRF.DSP/ACC1/TOTAL.OUT.DAC.QPARAM';
210 -  addr_ffi = 'TTF2.RF/LLRF.DSP/ACC1/FF_TABLE.IPARAM';
211 -  addr_ffq = 'TTF2.RF/LLRF.DSP/ACC1/FF_TABLE.QPARAM';
212 -  addr_spi = 'TTF2.RF/LLRF.DSP/ACC1/SP_TABLE.IPARAM';
213 -  addr_spq = 'TTF2.RF/LLRF.DSP/ACC1/SP_TABLE.QPARAM';
214 -  addr_sg  = 'TTF2.RF/LLRF.DSP/ACC1/GT_SYS_GAIN';
215 -  addr_lg  = 'TTF2.RF/LLRF.DSP/ACC1/GT_LOOP_GAIN';
216 -  addr_lop = 'TTF2.RF/LLRF.DSP/ACC1/LOOP_PHASE.PHASE';
217 -  addr_del = 'TTF2.RF/LLRF.DSP/ACC1/SP_DELAY_TIME';
218 -  addr_sfd = 'TTF2.RF/LLRF.DSP/ACC1/SP_FF_DELAY_TIME';
219 -  addr_fil = 'TTF2.RF/LLRF.DSP/ACC1/SP_FILL_TIME';
220 -  addr_dur = 'TTF2.RF/LLRF.DSP/ACC1/SP_FLAT_TOP_DUR';
221 -  addr_buf = 'TTF2.RF/ADC/GUN.PFOR/CURRENT';
222 -  addr_ofi = 'TTF2.RF/LLRF.DSP/ACC1/FF_I_OFFSET';
223 -  addr_ofq = 'TTF2.RF/LLRF.DSP/ACC1/FF_Q_OFFSET';
224 -  addr_ffm = 'TTF2.RF/LLRF.DSP/ACC1/FF_TABLE.MIX';
225 -  addr_fb  = 'TTF2.RF/LLRF.DSP/ACC1/OPEN_CLOSE_LOOP';
226 -  addr_ff  = 'TTF2.RF/LLRF.DSP/ACC1/FF_MODE';
227 -  addr_rat = 'TTF2.RF/LLRF.DSP/ACC1/FF_RATIO';
228 -  addr_bcmp= 'TTF2.RF/LLRF.DSP/ACC1/BEAM_COMP_ON_OFF';
229 -  addr_sp  = 'TTF2.RF/LLRF.DSP/ACC1/SP_VOLTAGE_MV';
230 -  addr_fam = 'TTF2.RF/LLRF.DSP/ACC1/FF_AMPL_CAL_MV_HV';
231 -  addr_fab = 'TTF2.RF/LLRF.DSP/ACC1/FF_AMPL_CAL_HV_BIT';
232 -  addr_spp = 'TTF2.RF/LLRF.DSP/ACC1/SP_PHASE_REL_BEAM';
233 -  addr_po  = 'TTF2.RF/LLRF.DSP/ACC1/SP_PHASE_OFFSET';
234 -  addr_ffp = 'TTF2.RF/LLRF.DSP/ACC1/FF_SP_PHASE_OFFSET';
235 -  addr_fri = 'TTF2.RF/LLRF.DSP/ACC1/REF_FF_TABLE.IPARAM';
236 -  addr_frq = 'TTF2.RF/LLRF.DSP/ACC1/REF_FF_TABLE.QPARAM';
237 -  addr_frm = 'TTF2.RF/LLRF.DSP/ACC1/REF_FF_TABLE.MIX';
238 -  addr_fru = 'TTF2.RF/LLRF.DSP/ACC1/REF_FF_TABLE.USR';
239 -  addr_ta3 = 'TTF2.DIAG/TIMER/MASTER/CH01.DELAY';
240 -  addr_ta6 = 'TTF2.DIAG/TIMER/MASTER/CH04.DELAY';
241 -  addr_dsd = 'TTF2.RF/TIMER/DSP1/CH01.DELAY';
242 -  addr_add = 'TTF2.RF/TIMER/ACC1/CH02.DELAY';
243 -  addr_igctstop = 'TTF2.RF/TIMER/KLY5DSP/CHANNEL.6.DELAY';
244 -  addr_spd = 'TTF2.RF/LLRF.DSP/ACC1/SP_DELAY_TIME';
245 -  addr_bed = 'TTF2.UTIL/LASER/GUN/T_DELAY12';
246 -  addr_nbu = 'TTF2.UTIL/LASER/GUN/PULSE_NUM';
247 -  addr_lat = 'TTF2.UTIL/LASERLINE/LASER.ATTENUAT/MOTOR.POS.SETPOINT';
248 -  addr_tor = 'TTF2.DIAG/TOROID/3GUN/CH00.DIFF_TD';
```

config_acc1 | Ln 228 Col 23 | OVR

…all RF-stations have identical algorithms with individual config-files.

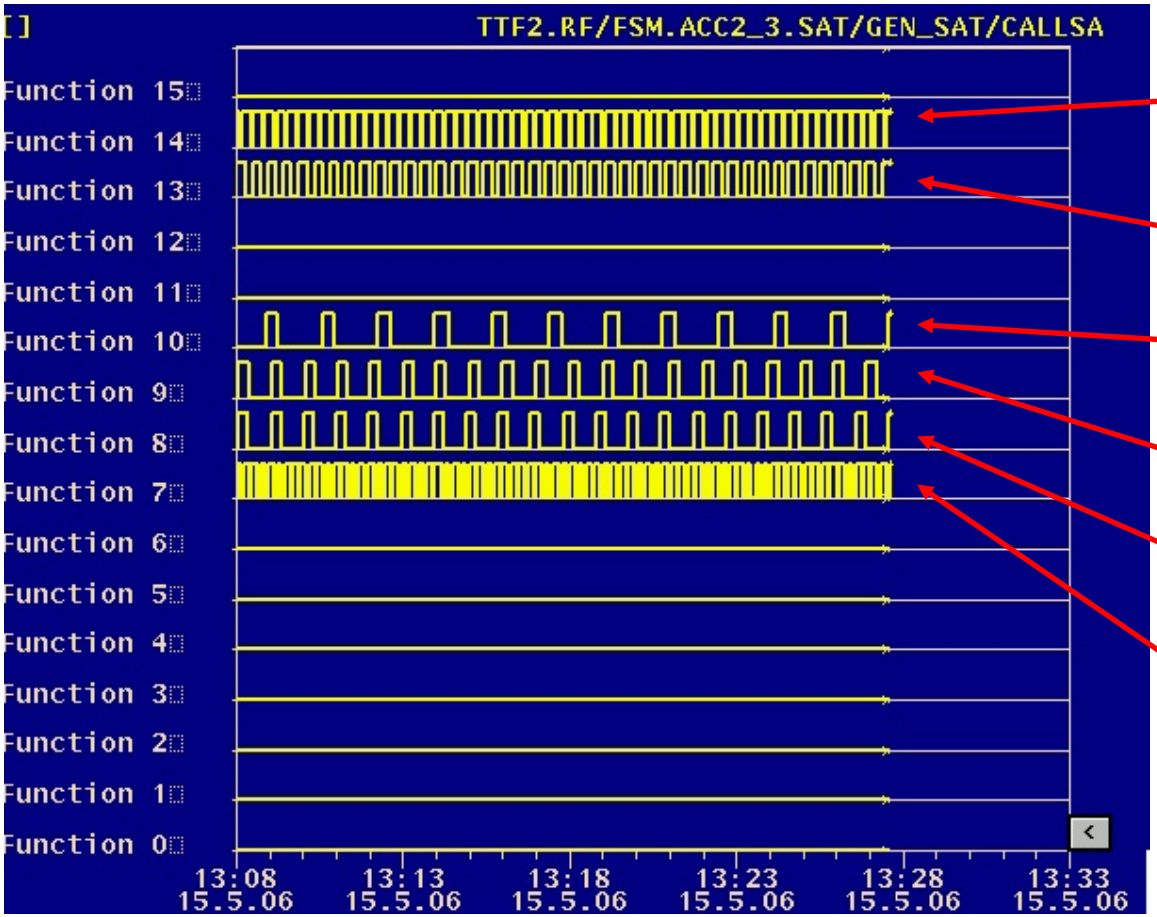
Procedure Server

Hist 0-15
Hist 16-31

``Stateless Procedures''-Server

	Configuration	Timeou	Short Description	Human-Readable Result	Numerical Res.		
FUN0	llrf/runDummy.csh out0.dat llrf/out0.dat	1500		Operation Successfull!	0	1.96	0.28
FUN1	/home/doocs/solaris2/obj/server/ttf/klys_fsm/runKlysFsm llrf/out1.dat	500		Script execution error.	-99	0	0
FUN2	llrf/runOffCal.csh out2.dat llrf/out2.dat	1500	Correcting Offsets.	Script execution error.	-99	0	0
FUN3	llrf/runSetSCandLP.sh out3.dat llrf/out3.dat	1500	Correcting loopphase.	Corrected loop phase.	2	-3.84	0.12
FUN4	llrf/runOneStepAFFDSP.csh out4.dat llrf/out4.dat	1500	Adapting Feedforward (DSP-par.).	Applied zero Feedforward (Feedforward was off)	0	0	0
FUN5	llrf/runOneStepAFF.sh out5.dat llrf/out5.dat	1500	Adapting Feedforward	Feedback is off. No FF-adaption possible.	0	0	0
FUN6	llrf/runDefaultFF.csh out6.dat llrf/out6.dat	1500	Applying default Feedforward.	Successfully set default feedforward.	0	0	0
FUN7	llrf/runCouplerRst7.sh out7.dat 1 llrf/out7.dat	1500	Checking coupler interlocks.	Executing...	-98	0	0
FUN8	llrf/runSetSCandLP.sh out8.dat 1 llrf/out8.dat	1500	Checking loopphase.	No drive from llrf is applied.	0	0	0
FUN9	llrf/runFieldQuality.sh out9.dat llrf/out9.dat	1500	Checking data quality.	FB=0 or FF=0! Data quality is unacceptable.	9	inf	17.42
FUN10	llrf/runCheckNetwork7.sh out10.dat llrf/out10.dat	1500	Checking network.	Network works fine.	0	0	0
FUN11	llrf/runRampFB.csh out11.dat llrf/out11.dat	1500	Ramping Feedback.	Successfully ramped FB to target value!	0	0	0
FUN12	llrf/runSetSPVal.csh out12.dat llrf/out12.dat	1500	Setting amplitude and phase.	Successfully ramped SP to target value!	0	0	0
FUN13	llrf/runOperatorAction.sh out13.dat llrf/out13.dat	1500	Looking for operator action.	Executing...	-98	0	0
FUN14	llrf/runKathLaser.sh out14.dat llrf/out14.dat	1500	Checking cathode laser settings.	Executing...	-98	0	0
FUN15	llrf/runGetDSPCalData.csh out15.dat llrf/out15.dat	1500	Retrieving DSP calibration-data.	Script execution error.	-99	0	0
FUN16	llrf/runEvalDSPCalData7.csh out16.dat llrf/out16.dat	1500	Evaluating DSP-Cal-Data.	Script execution error.	-99	0	0
FUN17	llrf/runSignalCalib7.sh out17.dat llrf/out17.dat	1500	Calculating cavity signal calibration.	There is no drive at all!	2	0	0
FUN18	llrf/runGetDetunings.csh out18.dat llrf/out18.dat	1500	Calculating detunings.	Calculating detuning done.	0	0	0
FUN19	llrf/runSetSPVal.csh out19.dat llrf/out19.dat	1500	Setting amplitude and phase (slowly).		0	0	0
FUN20	llrf/runKlyIsDown.sh out20.dat llrf/out20.dat	1500	Looking if Klystron is alive.	Executing...	-98	0	0
FUN21	llrf/runFindFancyPulses.sh out21.dat llrf/out21.dat	1500	Looking for fancy pulses.	Feedforward is off.	0	0	0
FUN22	offCal ACC456 out22.dat out22.dat	0		Script execution error.	-99	0	0
FUN23	setSCandLP ACC456 out23.dat out23.dat	0		Calculated bandwidth out of range. (Maybe there is no drive signal at all?)	-99	0	0
FUN24	llrf/runRippleLoad.sh out24.dat llrf/out24.dat	500	Applying ripple-correction.	Script execution error.	-99	0	0
FUN25	llrf/runRippleLoad.sh out25.dat 0 llrf/out25.dat	500	Restoring zero ripple-table.	Script execution error.	-99	0	0
FUN26	defaultFF_ACC456 out26.dat out26.dat	0		Script execution error.	-99	0	0
FUN27		0			0	0	0
FUN28	./runfsmklys4 out28.txt out28.txt	0		Script execution error.	-99	0	0
FUN29	./runfsmklys5 out29.txt out29.txt	0		Script execution error.	-99	0	0

Procedure History



Check Cathode Laser

Check Operator Action

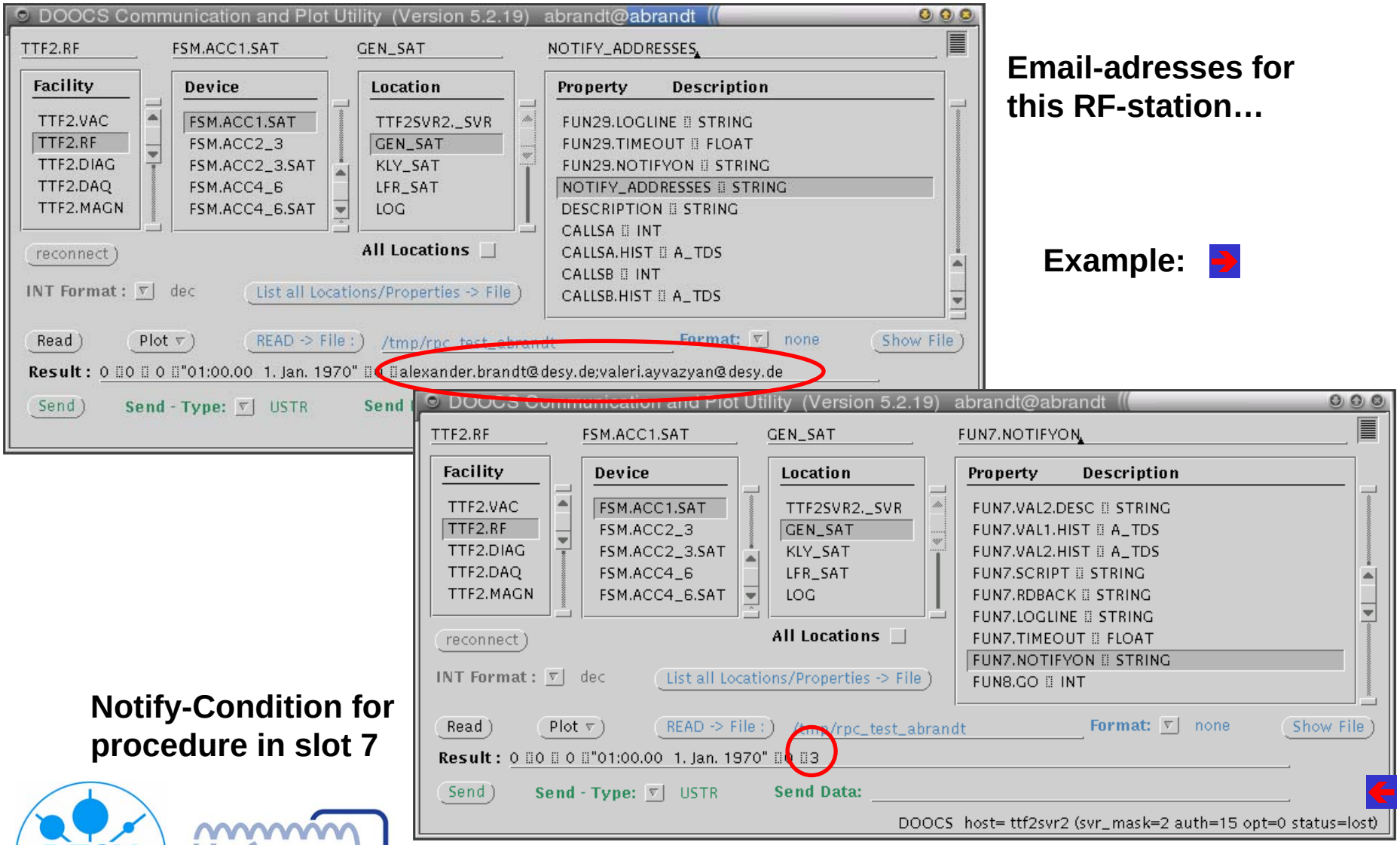
Check config file (network availability)

Monitor data quality

Check for loop phase

Check for coupler interlocks

Email-Notification

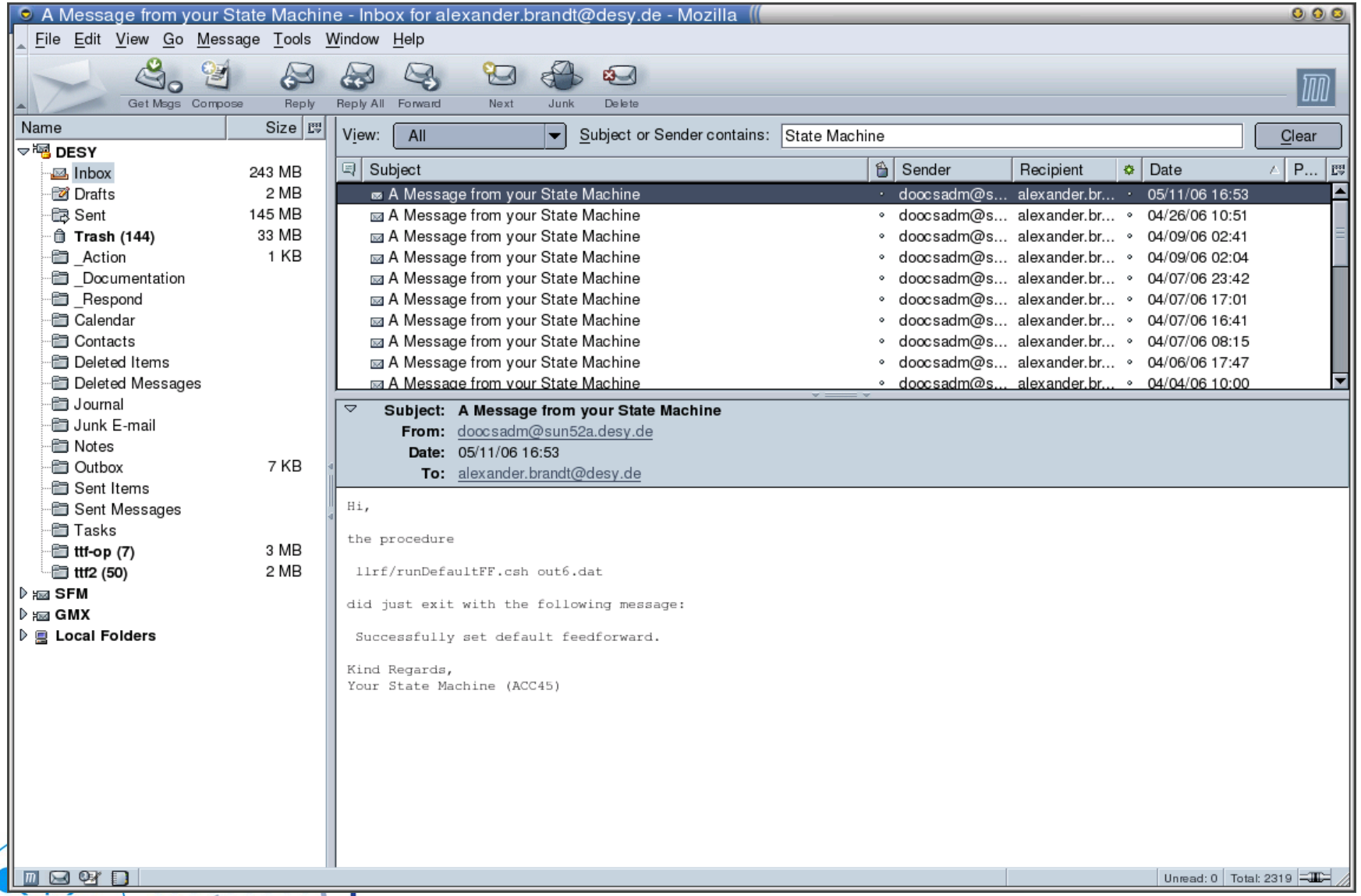


Email-adresses for this RF-station…

Example:

Notify-Condition for procedure in slot 7

Email-Example



On-Line Reconfigurability

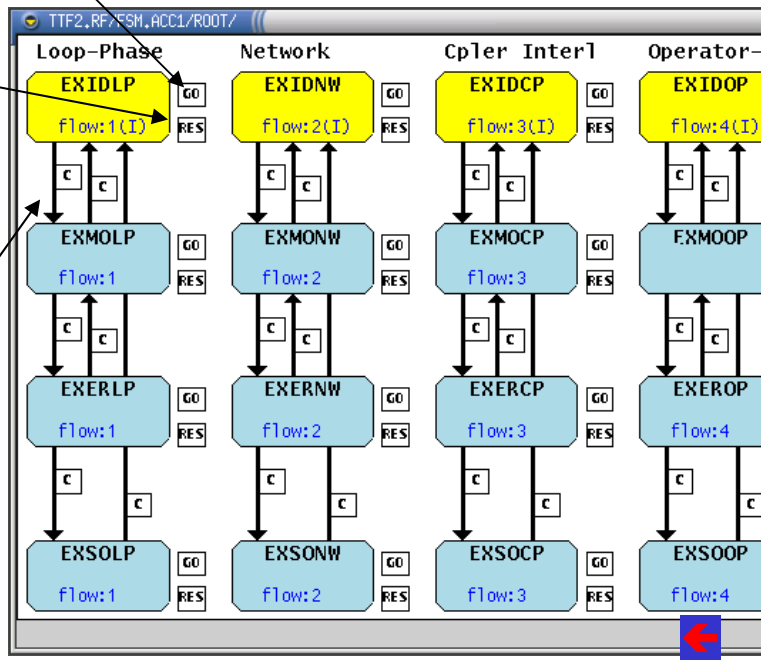
A click here defines what happens on enter…

- **can be a number (like ‘wait 25 pulses’),**
- **can be an event (make transition somewhere),**
- **can be a procedure**
- **can be a combination of them.**

Where to find the numerical result once the procedure (timer) has finished - can be any DOOCS-address!

Conditions, that the result is compared to (which lead to a transition)

- **can be (a list of) number(s)**
- **can be a ‘not’, ‘else’, ‘all’**



Off-Line Restructurability

emacs@

File Edit Options Buffers Tools C++ Help

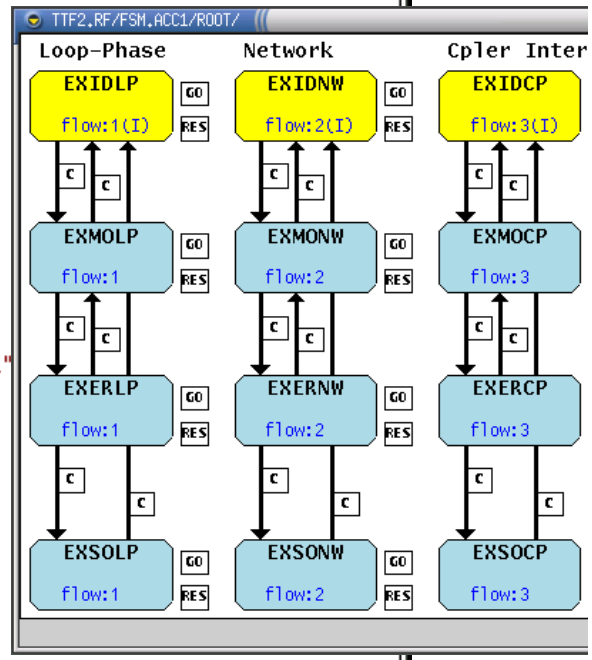
```
//// DO NOT EDIT THIS AND NEXT TWO ROWS
// init function for state 'EXERDQ'
int     ROOT_EXERDQ_init_action(FSMstate *st_p_)
{

        fputs("ROOT_EXERDQ:: \"init\" event  procedure: ROOT_EXERDQ_init_action \n", stderr);

        TStateLogic* sd=TStateLogic::factory("EXERDQ");
        sd->set_go("ROOT_EXCEPTION.EXERDQ_GO");
        sd->set_res("ROOT_EXCEPTION.EXERDQ_RES");
        sd->add_event("EXERDQ", "exerdq_exmodq", "ROOT_EXCEPTION.EXERDQ_T1");
        sd->add_event("EXERDQ", "exerdq_exsodq", "ROOT_EXCEPTION.EXERDQ_T2");

        return ST_OK;
}

//// DO NOT EDIT THIS AND NEXT TWO ROWS
// enter function for state 'EXERDQ'
int     ROOT_EXERDQ_enter_action(FSMstate *st_p_)
{

        fputs("ROOT_EXERDQ:: \"enter\" event  procedure: ROOT_EXERDQ_enter_action \n"

        TStateLogic* sd=TStateLogic::factory("EXERDQ");
        sd->enter(st_p_);

        return ST_OK;
}

//// DO NOT EDIT THIS AND NEXT TWO ROWS
// during function for state 'EXERDQ'
int     ROOT_EXERDQ_during_action(FSMstate *st_p_)
{

        fputs("ROOT_EXERDQ:: \"during\" event  procedure: ROOT_EXERDQ_during_action \n", stderr);

        TStateLogic* sd=TStateLogic::factory("EXERDQ");
        sd->during(st_p_);

        return ST_OK;
}
```

init

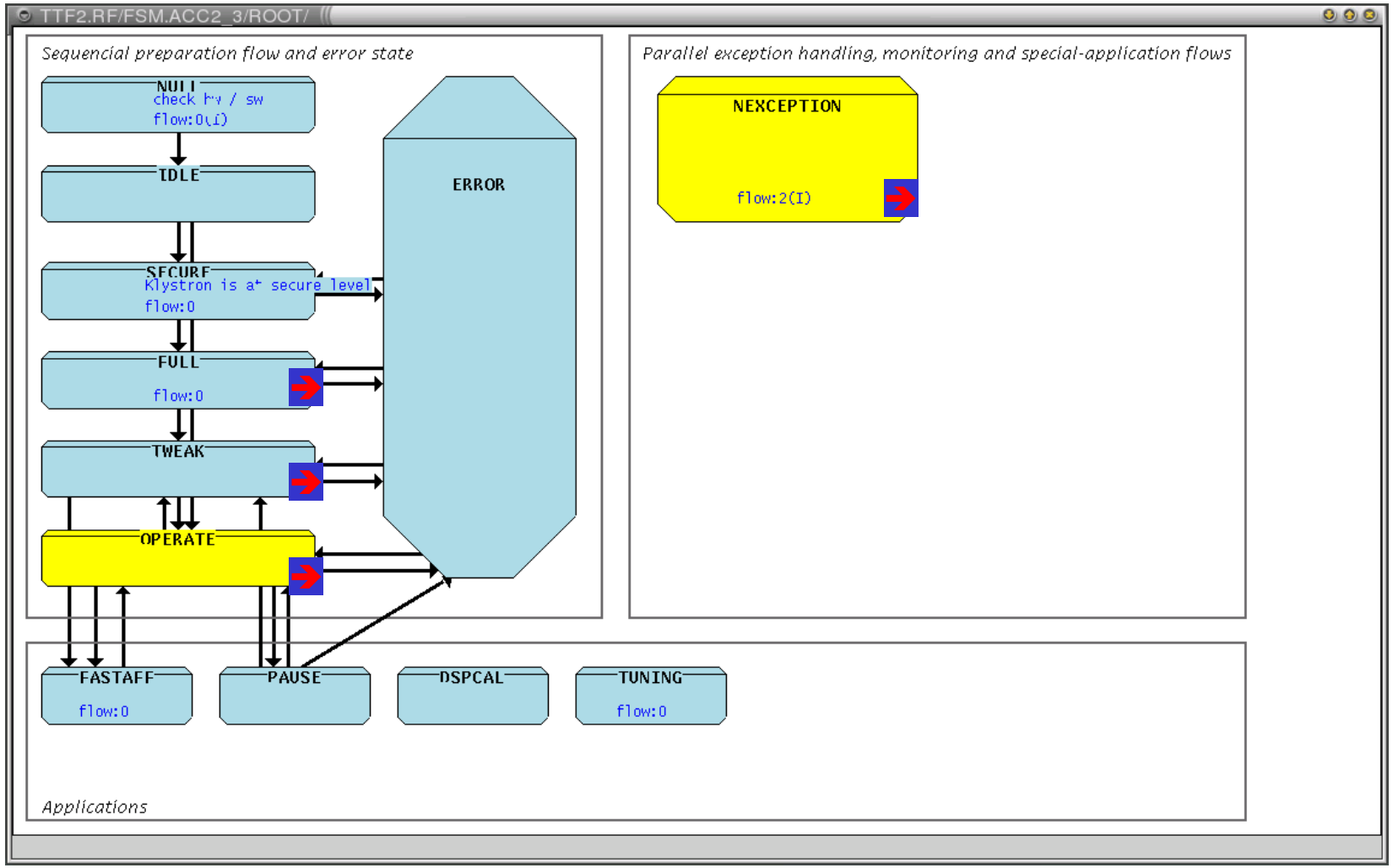
enter

during

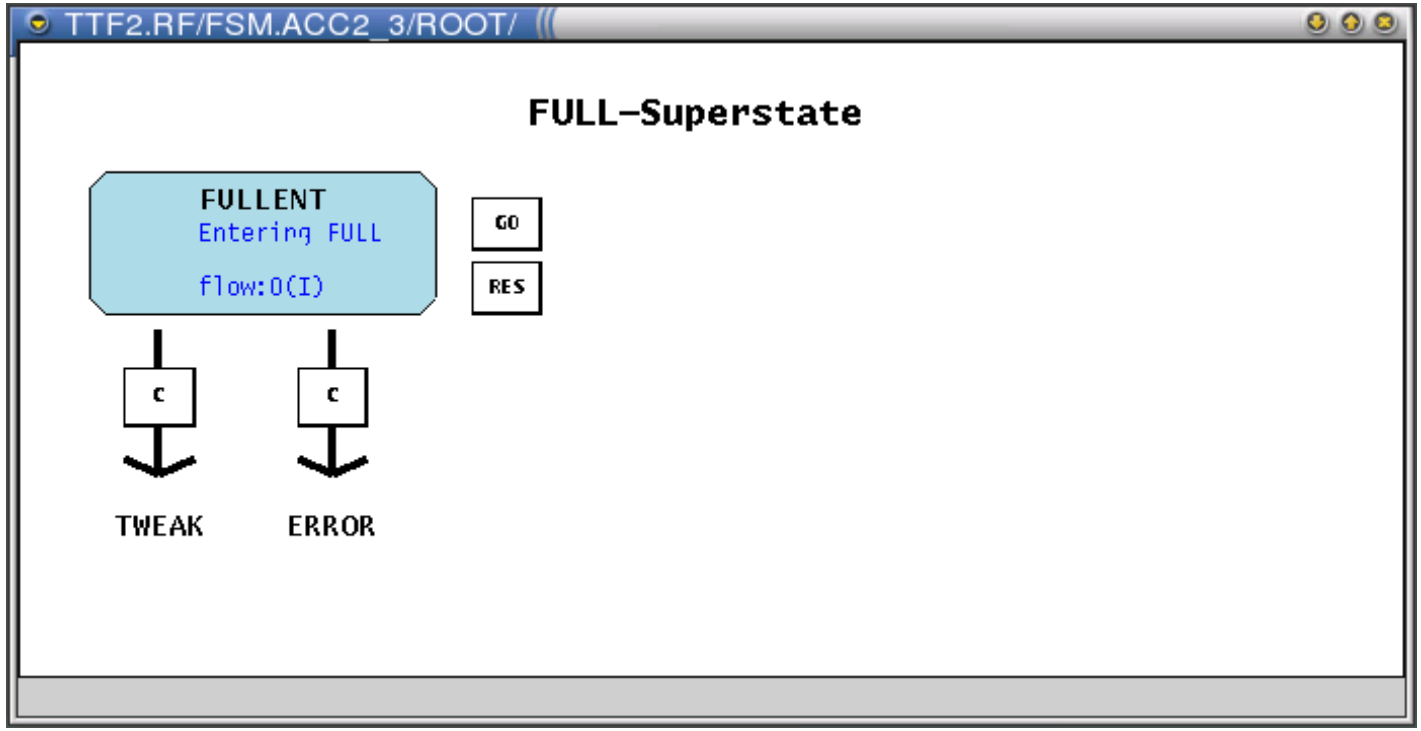
--:** llrf_fsm_panel_ROOT.cc (C++ Abbrev)--L1949--94%



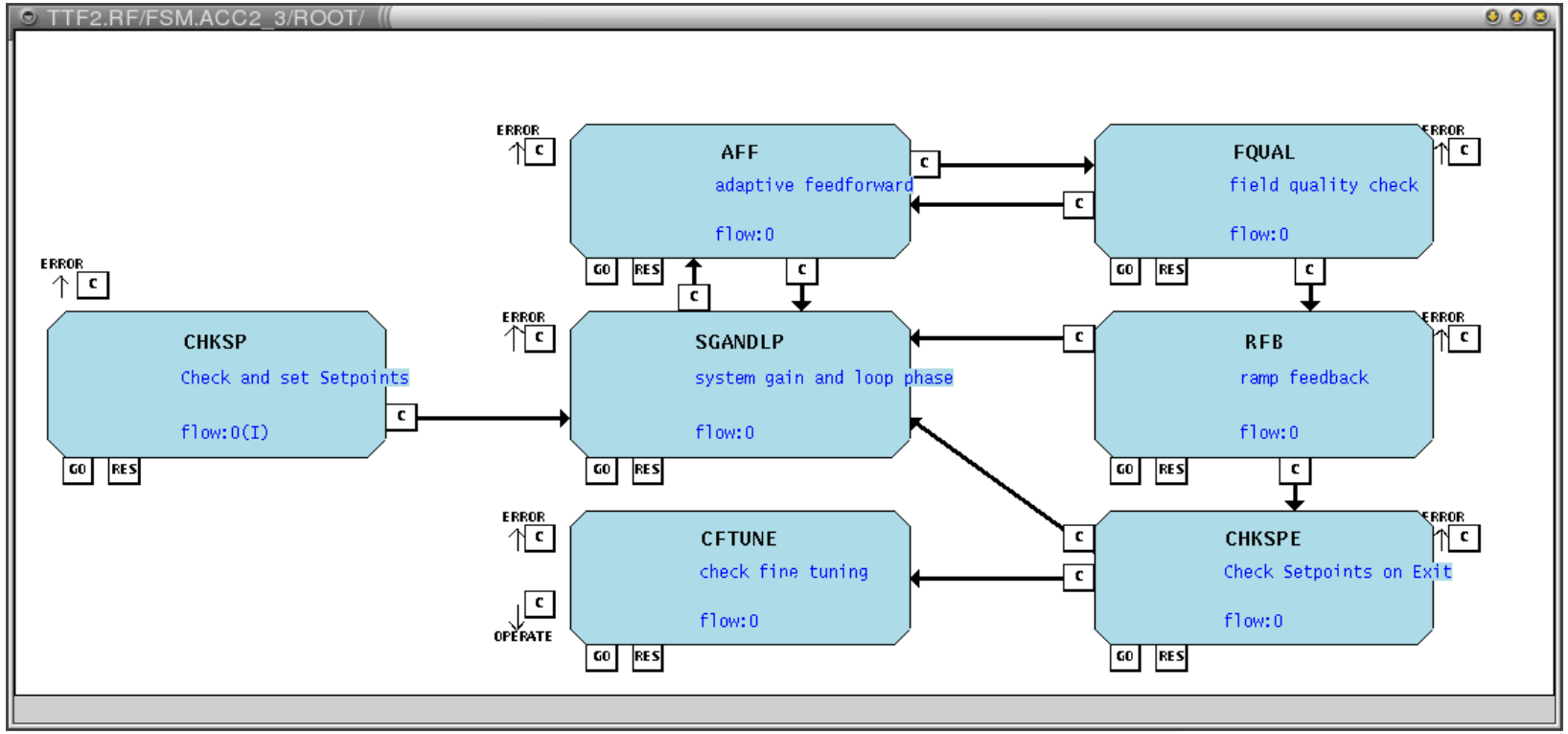
FSM Top Level View



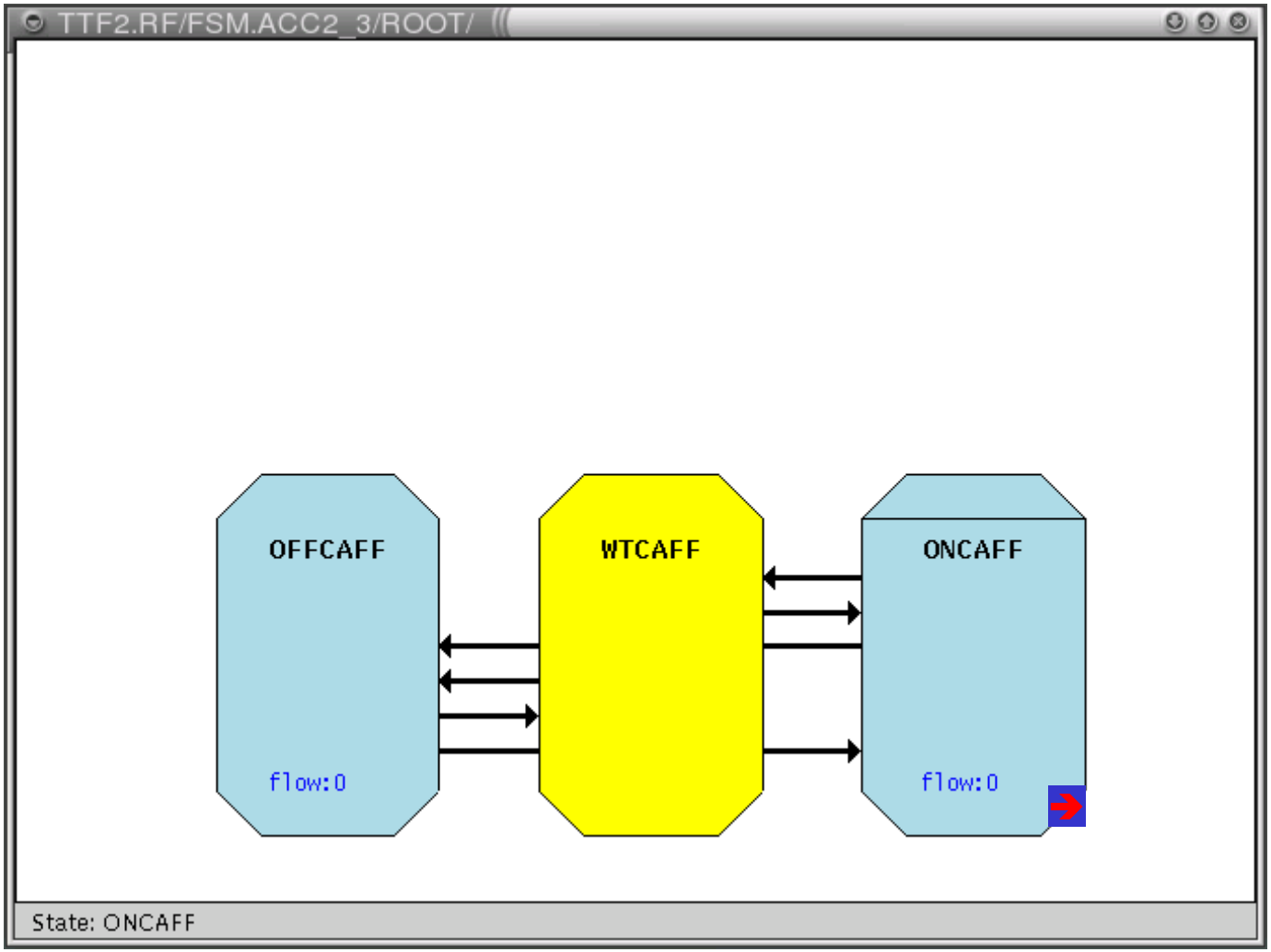
FSM Full State



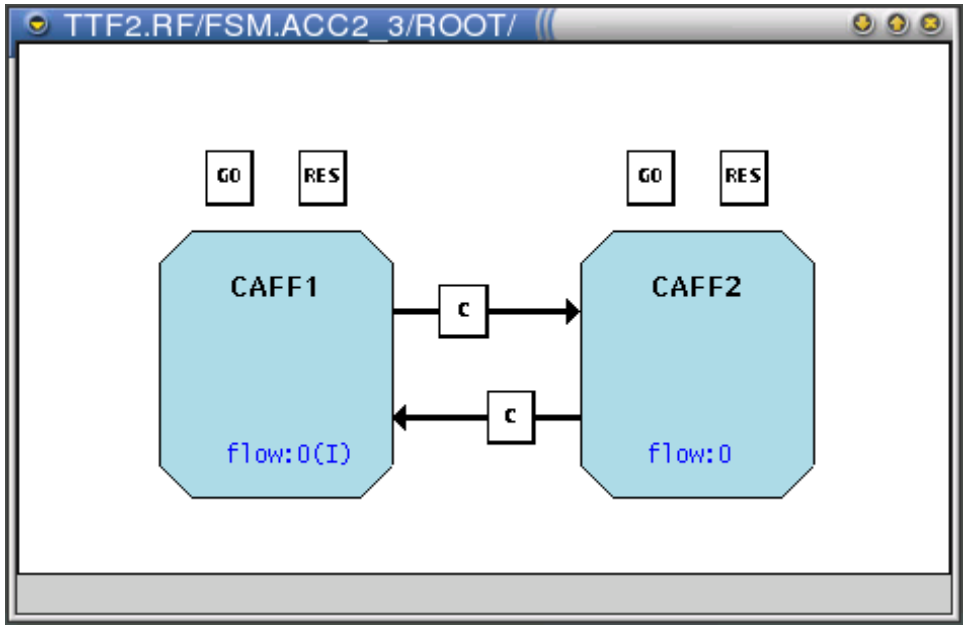
FSM Tweak State



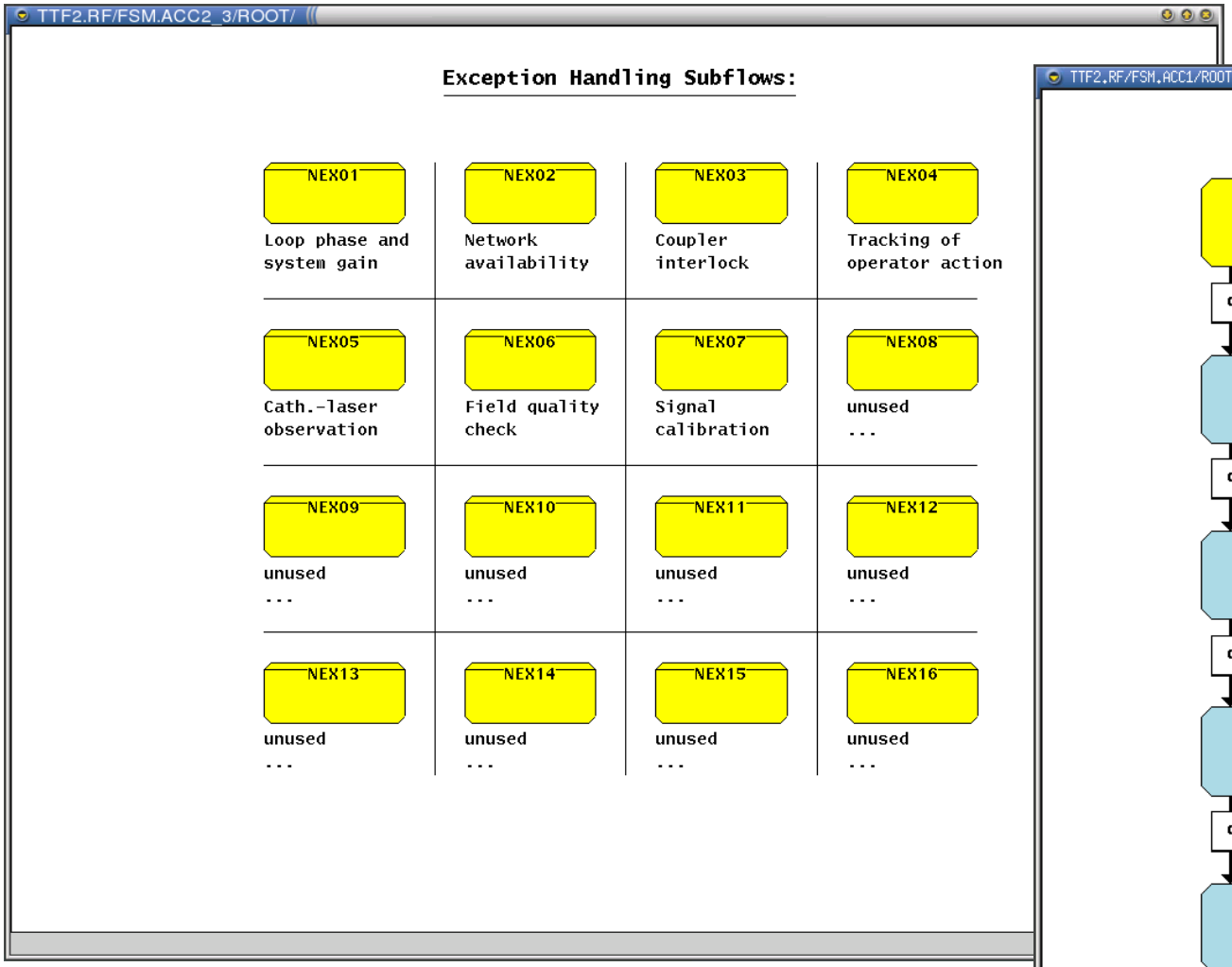
FSM Operate State

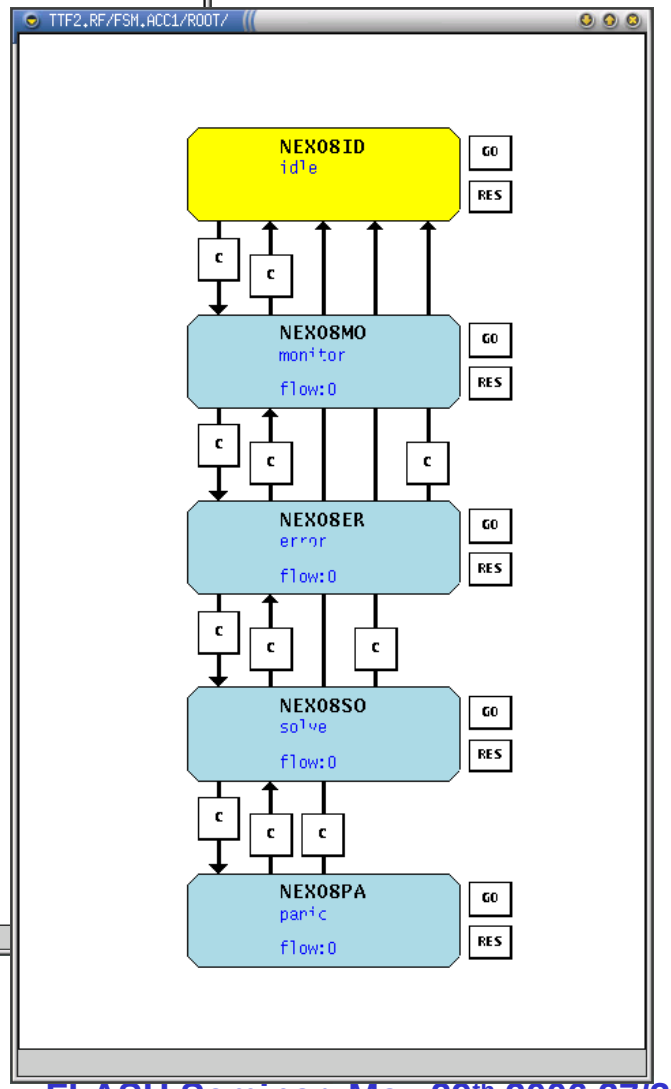


FSM ONCAFF State



FSM Exception State



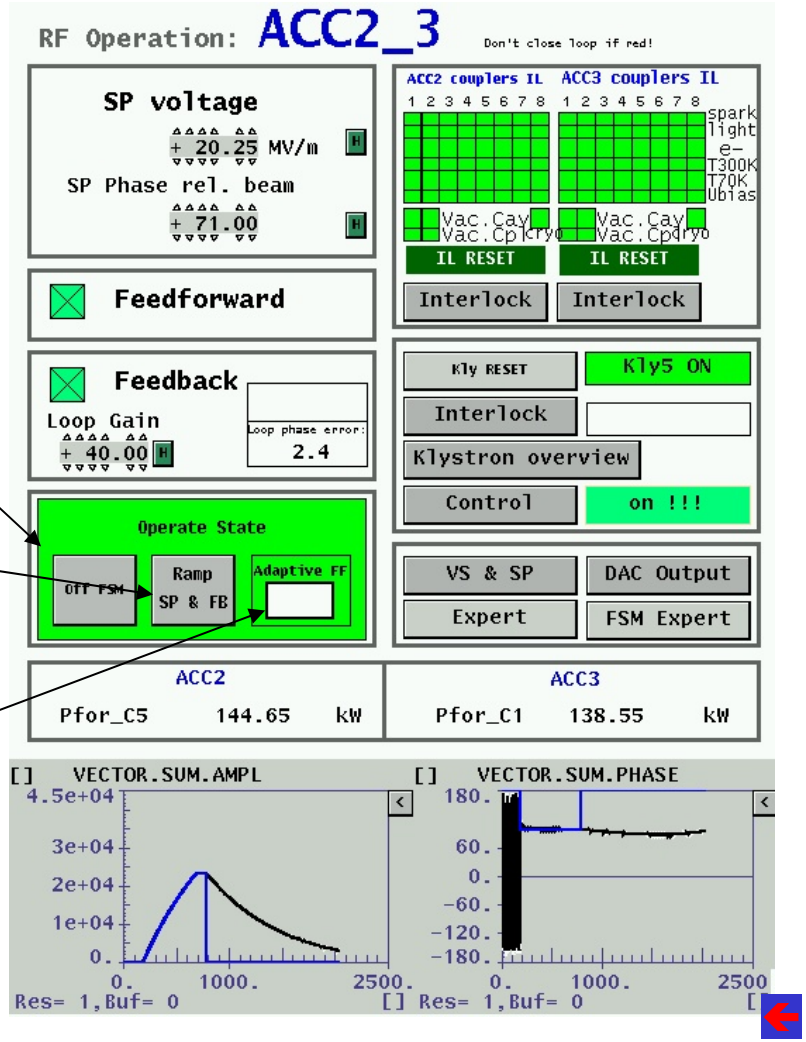


Simple Operators View

Switch off FSM

Ramp RF to the target values (…)

Enable adaptive FF



FSM-Expert View

Checkboxes determine the behavior of the algorithms (and the algorithms determine the behavior of the FSM).

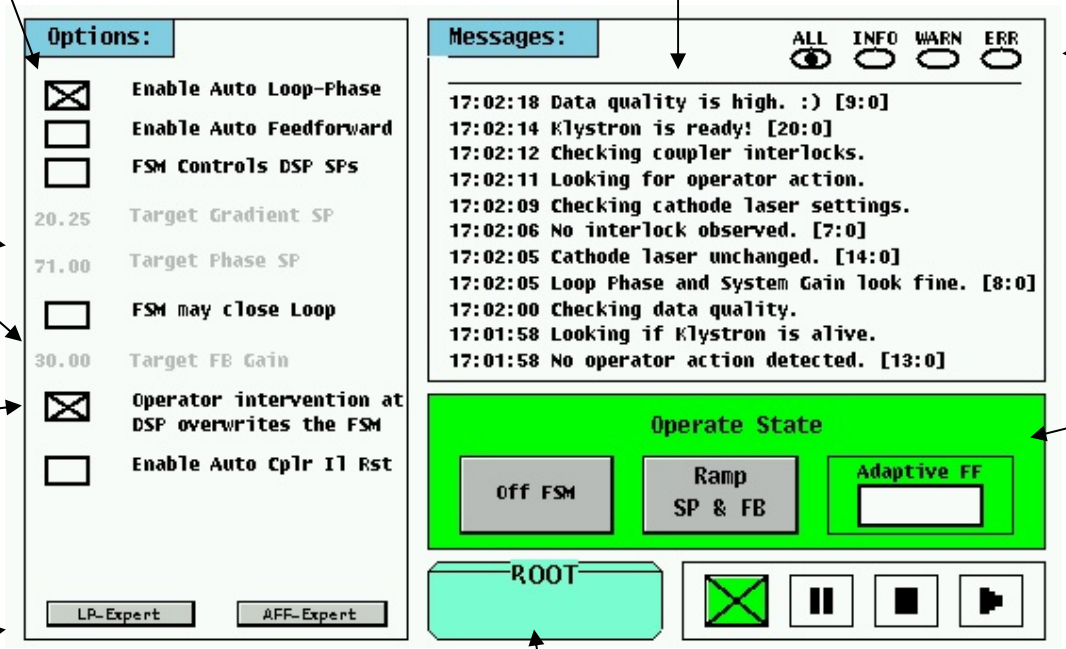
Scrolling messages from all procedures

Debug level

Target values are automagically updated while operator is working with the DSP!

…but only as long as this is desired!

Simple operators view



Algorithm expert settings

Access to FSM-structure (for on-line reconfig)

Operation Experience

- "Permanent features"
  - Loop phase running with <1 error/week (fixed usually the next day)
  - Adaptive feedforward (rarely used) with ~1 error/week (fixed next day)
  - Few things run without operator awareness (signal calibration, operator-tracking)
- "Occasional features"
  - DSP ramp up after down: tested but not used
  - DSP ramp down after klystron trip: tested but not used
- "Optional Features"
  - DSP calibration, detuning display, … just used by me from time to time

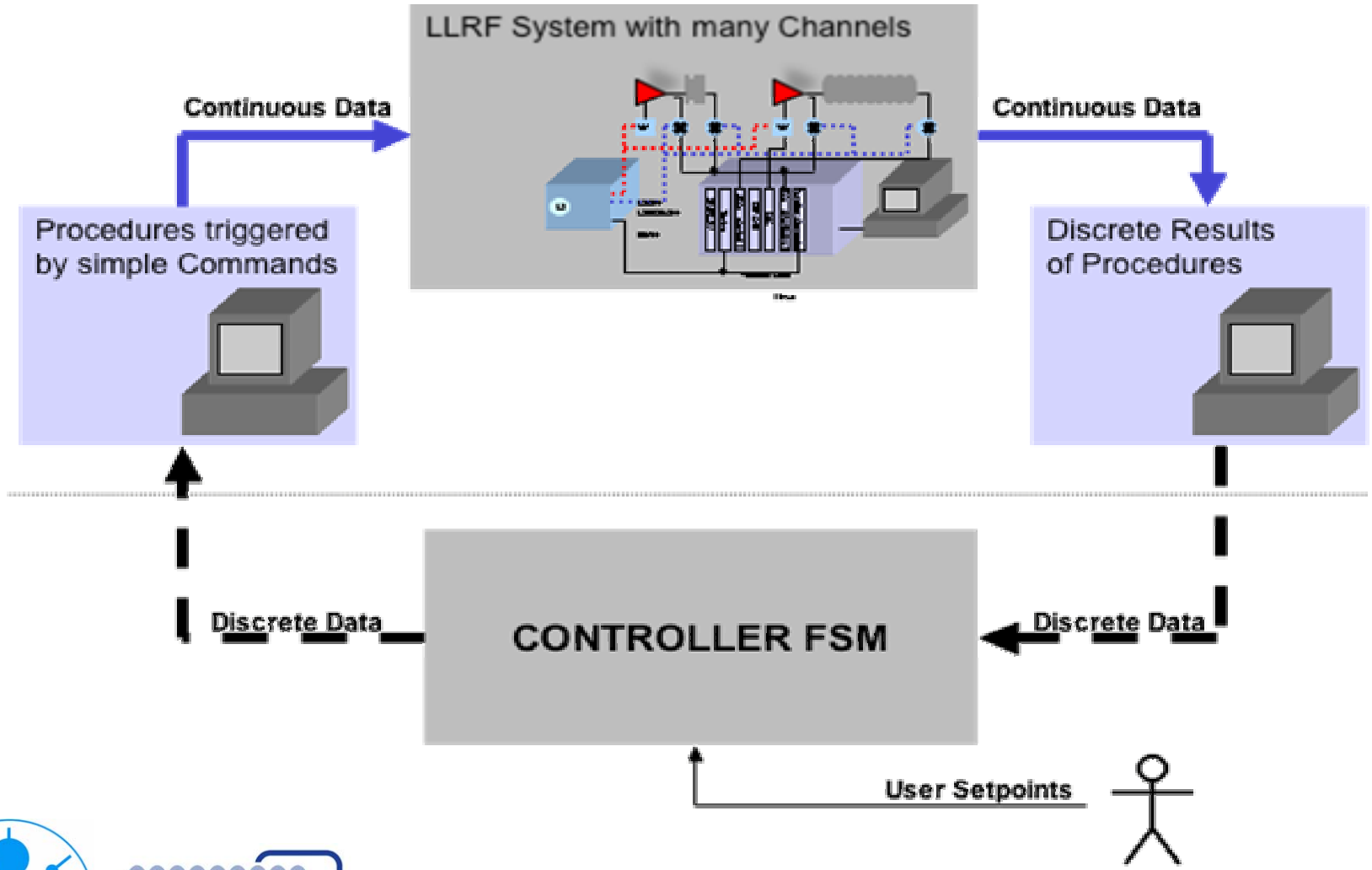
DDD FSM Framework

- Nice, but probably to unflexible
- That's why my implementation extended it by
  - Procedure server (with features like timeout and email notification)
  - Factory-classes for a simple reconfiguration (still needs compilation)
  - Parser that interprete online configuration ("go", "res" and "c" - buttons)
- Re-working the automation should be as easy as re-designing a panel
- Extension of the DDD FSM Framework is in progress…

Finally…

- **FSM is in permanent commissioning - Valeri and I are activating features and testing them**
- **Adaptive Feedforward is**
  - **available at FLASH**
  - **tested at SNS (to be implemented in August)**
  - **on it's way into the FPGA**
- **Still: there is a lack of operator acceptance / cooperation**
- **Good experience made with the flexibility of this approach (fast bug-fixing, adaptation to operator needs)**

...


LLRF System with many Channels
Continuous Data
Continuous Data
Procedures triggered by simple Commands
Discrete Results of Procedures
Discrete Data
CONTROLLER FSM
Discrete Data
User Setpoints