EISCAT Svalbard Radar

Radar Controller Programming:
RACELAN
Instruction Sequences
Timing Conditions

Terrance Ho & Reinhard Hund

June 05, 1997

version of this document: 1.6/ June 05, 1997
WWW: please look at
http://www.eiscat.uit.no/EISCAT/Hardware/ESR/html/documentations.html
or send e-mail to reinhard.hund@bigfoot.com or ho@linax1.dnet.gwdg.de

Contents

1 Radar Controller Programming

1.1 Default Bit Pattern - Transmitter

Using the RACELAN compiler, the TX radar controller bit pattern is set to 07FBFFF8 (hex) as default. This corresponds to the following:

bit	default value	status
0	0	receiver protector: unprotected (open) and exciter mixer: off
1	0	pre-amplifier: on
2	0	noise calibrate: off
3	1	spare (used for oscilloscope triggering)
4	1	mixer control
5	1	FSEL0 inactive
6	1	FSEL1 inactive
7	1	FSEL2 inactive
8	1	FSEL3 inactive
9	1	UNIT0 inactive
10	1	UNIT1 inactive
11	1	UNIT2 inactive
12	1	ALL inactive
13	1	OPER inactive
14	1	WREG inactive
15	1	FLOAD inactive
16	1	MOSEL inactive
17	1	RFDR inactive
18	0	PFLIP set to 0 deg
19	1	spare (bit set)
20	1	spare (bit set)
21	1	spare (bit set)
22	1	spare (bit set)
23	1	spare (bit set)
24	1	spare (bit set)
25	1	spare (bit set)
26	1	spare (bit set)
27	0	beam off
28	0	sampling off
29	0	antenna 0 not selected
30	0	antenna 1 not selected
31	0	sync bit not set

Please note that this default setup must correspond with the default setup generated by the patch panel crate which is connected to the TX radar controller. Having different patterns on the hardware side will cause spikes at the end of the integration period. The default bit pattern is defined in file hardware.dly.

1.2 Default Bit Pattern - Receiver

The default bit pattern of the RX radar controller generated by RACELAN is C007FC00. This can be changed in file hardware.dly.

1.3 Limits - Assorted List

- A maximum of 80 characters in a line is allowed for a RACELAN program.
- A TAB characters may confuse the compiler
- The maximum time for an instruction to last is 3 s = 3,000,000 μs.
- The radar controller memory is 256kB long.
- The compiler doesn't accept a TXBITON, TXBITOFF, RXBITON or RXBITOFF along the line with other instructions than these.
- There must be at least 0.3 usec time difference from the previous instruction to the END instruction in order to insert the three last lines of code for the END instruction.

2 RACELAN Description

2.1 Introduction

The RACELAN compiler reads an input file containing the transmitter and/or receiver instructions and creates two sets of files for each radar controller. The files for the TX radar controller have the extension .tasc and tbin. The .tasc file contains the TX radar controller commands in hex code and can be read using a text editor. This file can be used for debugging purposes. The .tbin file is a binary file which is loaded into the TX radar controller. Similarly, there are two files for the RX radar controller with the extensions .asc and .bin.
The compiler is case insensitive. Therefore, uppercase and lowercase are treated as the same. Each field in a line must be separated by at least one blank or a comma. All times are given in μsecs resolution.

Starting the RACELAN Compiler The compiler version 3.0 + allows the suppression of RX and / or TX output file, the specification of the input filename and the suppression of the hardware delay checking (for test purposes only!).

```
eiscat1% racelan
*** RACELAN COMPILER FOR ESR TX AND RX RADAR CONTROLLERS ***
*** VERSION 3.0  CREATED ON 15/5/97 WRITTEN BY T.HO ***
THE COMMAND IS:  racelan -ffilename -nd -t -r
WHERE -f   is the input filename
      -nd  is no hardware delay checking
      -t   create TX output files
      -r   create RX output files
```

Note: The environment variable RACELAN must point to the directory in which the file hardware.dly with the hardware delay times is located.

2.2 General Instructions

The general syntax to control a radar controller bit is *AT time argument*, where *time* is given in μsec resolution. In order to achieve 100 nanosec resolution *time* is given as a real number, eg 100.1. The *argument* is either a transmitter or receiver bit mnemonic and is described in subsections 2.3 and 2.4. It is allowed to have several transmitter and/ or receiver mnemonics in a single instruction. If a combination of mnemonics is illegal the compiler will give an error message and abort.
There are also instructions for defining the status bits which are sent to the DSPs for DSP program control with the DEF instruction, looping of a block of instructions with the DO n and ENDDO instructions and dynamic control of the *time* in an AT instruction through the use of the variables SETTCR and INCTCR.

COMMENTS The % as the first character in a line is treated as a comment line. In-line comments are also allowed. Blank lines are also allowed as an aid to improve program readability.

SETTCR The SETTCR *time* instruction sets SETTCR to a time increment in μsecs. SETTCR is added to the *time* in an AT instruction and stays in effect until redefined[1]. The default is zero.

eg SETTCR 20000

INCTCR The INCTCR *time* instruction increments SETTCR by *time*. See also SETTCR.

eg INCTCR 1000 (SETTCR = SETTCR + 1000)

DO *n* / ENDDO The DO *n* instruction repeats a set of instructions *n* times up to the ENDDO instruction. The nesting of DO loops is not allowed.

eg
```
SETTCR - 35000
DO 7
INCTCR 35000
AT 7000 ...
AT 14000 ...
AT 21000 ...
AT 28000 ...
AT 35000 ...
ENDDO
```

The ENDDO instruction closes the DO loop.

[1] therefore SETTCR will usually have to be reset after a DO loop to continue with absolute AT *time* settings

END 'END' is handled as an argument in the AT *time* END instruction and defines the end of a program.
Hardware action: the radar controller is programmed with the following control word sequence: 80 (reload start address), 00 (no operation), 40 (end of program). The dwell time is 0 for each, the output word is kept.
There must be at least 0.3 usec time difference from the previous instruction to the END instruction in order to insert the three last lines of code for the END instruction.

2.3 Specific Instructions for the Radar Controller (TX)

NOTE: The hardware's execution time of any of the following instructions has to be obeyed. It is *not* checked by the RACELAN compiler. Look at subsection 4.1.

DEF MAXUNITNO *nmax* Defines with *nmax* the highest used DDS-unit (exciter) number in the program. This is of specific importance for the FLOAD UNIT* instruction. Since the exciter is equipped with four DDS units, the range for *nmax* is [0 ..3]. With six units later, the range will be [0 .. 5]. See also FLOAD.

AT *time arguments* The following arguments for the AT instruction (see 2.2) can be used to control radar controller (TX) bits:

WREG FSEL*n*, UNIT*m*, OPERA/B Write to exciter unit m (UNIT0 .. UNIT5) with register A or B (OPERA or OPERB) with frequency n (FSEL0 .. FSEL15). The order or FSELn, UNITm and OPER is not important.
Hardware action: The UNIT bits 9, 10 and 11 (UNIT0, UNIT1, UNIT2) or bit 12 (UNIT* for all units) are set accordingly to the requested unit number (all bits active low). Additionally bit 13 (OPER) is set high with OPERA or set low with OPERB. At the same time also bit 14 (WREG) is set low. WREG will be pulled high again after 100ns, the other bits remain unchanged.

FLOAD UNIT*n* / UNIT*, OPERA/B Load the exciter's phase increment register of unit n (UNIT0 .. UNIT5) or units [0 .. nmax][2] (UNIT*) with register A or B (OPERA or OPERB) . The order of UNITn / UNIT* and OPERA/B is not important.
Hardware action: The UNIT bits 9, 10 and 11 (UNIT0, UNIT1, UNIT2) or bit 12 (UNIT*) are set accordingly to the requested unit number (all bits active low). Additionally bit 13 (OPER) is set high with OPERA or set low with OPERB. At the same time bit 15 (FLOAD) is set low. FLOAD will be pulled high again after 100ns, the other bits remain unchanged.

MOSEL UNITn Select the exciter unit n (UNIT0 .. UNIT5).
Hardware action: The UNIT bits 9, 10 and 11 (UNIT0, UNIT1, UNIT2) or bit 12 (UNIT*) are set accordingly to the requested unit number (all bits active low). At the same time bit 16 (MOSEL) is set low. MOSEL will be pulled high again after 100ns, the other bits remain unchanged.

BEAMON Switch the beam on (controls the pulsers in the power amplifiers).
Hardware action: BEAMON sets bit 27 (BEAM) high.

BEAMOFF Switch the beam off (controls the pulsers in the power amplifiers).
Hardware action: BEAMOFF sets bit 27 (BEAM) low.

[2] c.f. DEF MAXUNITNO *nmax*

RFDRON Switch the RF on.
Hardware action: RFDRON sets bit 17 (RFDR) low.

RFDROFF Switch the RF off.
Hardware action: RFDROFF sets bit 17 (RFDR) high.

PHA0/180 Select 0° or 180° phase.
Hardware action: PHA0 sets bit 18 (PHASE) low, PHA180 sets bit 18 (PHASE) high.

CALON Switch the noise generator on for calibration. Noise will be injected into the receiver chain in front of the receiver protector.
Hardware action: CALON sets bit 2 (CAL) high.

CALOFF Switch the calibration noise off.
Hardware action: CALOFF sets bit 2 (CAL) low.

PREAMPON Switch the pre-amplifier on.
Hardware action: PREAMPON sets bit 1 (PREAMP) low.

PREAMPOFF Switch the pre-amplifier off and thereby sets it into protected state.
Hardware action: PREAMPOFF sets bit 1 (PREAMP) high.

RXPOFF Switch the receiver protector off, the TXSAMPLE switch off and the (second) mixer in the analog part of the exciter off.
Hardware action: RXPOFF sets bit 0 (RXPROT) low.

RXPON Switch the receiver protector on and thereby sets the receiver into the protected state. This also switches on the TXSAMPLE switch causing a sample of the transmitted signal to be fed into the receiver chain and switches on the (second) mixer in the analog part of the exciter.
Hardware action: RXPON sets bit 0 (RXPROT) high.

ADCTRIGON Switches the sampling signal for the A/D converters in the power amplifiers on.
Hardware action: ADCTRIGON sets bit 28 (ADCTRIG) high.

ADCTRIGOFF Switches the sampling signal for the A/D converters in the power amplifiers off.
Hardware action: ADCTRIGOFF sets bit 28 (ADCTRIG) low.

ANTENNA*n* Selects antenna *n*, (n = [0 .. 2])
Hardware action: ANTENNA*n* sets bits 29 and 30 (ANTENNA0, ANTENNA1). n=0: both bits low, n=1: bit 29 high and bit 30 low, n=2:bit 29 low and bit 30 high.

TXSYNCON Set the sync bit on.
Hardware action: TXSYNCON sets bit 31 (TXSYNC) high.

TXSYNCOFF Set the sync bit off.
Hardware action: TXSYNCOFF sets bit 31 (TXSYNC) low.

TXBITON [0] ... [,31] Set TX bit(s) on. There are **no** checks implemented, nor does an interpretation concerning active high/ active low signals take place. Use this instruction at your own risk and for radar controller testing purposes only. It is your responsibility if you damage the radar.

TXBITOFF [0] ... [,31] Set TX bit(s) off. There are **no** checks implemented, nor does an interpretation concerning active high/ active low signals take place. Use this instruction at your own risk and for radar controller testing purposes only. It is your responsibility if you damage the radar.

2.4 Specific Instructions for the Radar Controller (RX)

DEF The DEF instruction for the radar controller (RX) defines the status bits 0-255 (ie bits S0-S7) for the DSP interface board for DBV42_1 or DBV42_2 with a symbolic name (*SYMB_NAME*) of the user's choice, which when used with the AT *time SYMB_NAME* is sent to the appropriate DSP termed DBVS1 and DBVS2. It is imperative that each *SYMB_NAME* that is defined be unique. The following definitions apply to DBV42_1:

```
DEF   DBVS1_0      SYMB_NAME
      .
      .
DEF   DBVS1_255    SYMB_NAME
eg              DEF DBVS1_1 STC_WRT1.
```

The following definitions apply to DBV42_2:

```
DEF   DBVS2_0      SYMB_NAME
      .
      .
DEF   DBVS2_255    SYMB_NAME
eg              DEF DBVS2_1 STC_WRT2.
```

AT *time arguments* The following arguments for the AT instruction (see 2.2) can be used for controlling the radar controller (RX) bits:

SYMB_NAME As defined above. Note, each *SYMB_NAME* must be unique. Hardware action: The bits 0 ..7 (S0 .. S7) are set according to the DEF instruction. At the same time bit 8 (INT1) or 9 (INT2) is set high (see DEF). This bit will be reset again after 100ns (strobed bit).

ENABMn [,ENABMm] [, ...] ENABM1 [,ENABM2] [,ENABM3] [,ENABM4] [,ENABM5] [,ENABM6] enables data writing to buffer memory 1 2 3 4 5 6 respectively. The order of the buffer memories is not important.
Hardware action: ENABMn sets bit (n+9) i.e. one of the bits 10 .. 15 (CHON 1 .. CHON 6) low.

DISBMn [,DISBMm] [, ...] DISBM1 [,DISBM2] [,DISBM3] [,DISBM4] [,DISBM5] [,DISBM6] disables data writing to buffer memory 1 2 3 4 5 6 respectively. The order of the buffer memories is not important.
Hardware action: DISBMn sets bit (n+9) i.e. one of the bits 10 .. 15 (CHON 1 ..CHON 6) high.

BUFFLIP1 Flip the buffer memory (1-3) sides.
Hardware action: BUFFLIP1 sets bit 17 (Bufflip I) low. The bit will be set high again after 100ns (strobed).

BUFFLIP2 Flip the buffer memory (4-6) sides.
Hardware action: BUFFLIP2 sets bit 18 (Bufflip II) low. The bit will be set high again after 100ns (strobed bit).

SETCOUNT Set the address counters.
Hardware action: SETCOUNT sets bit 16 (Set-count) low. The bit will be set high again after 100ns (strobed bit).

NCOSELn Select NCO frequency *n*. *n* must be between 0 and 1023.

eg NCOSEL33 select NCO frequency 33

Hardware action: NCOSELn sets bits 19 .. 28 (NCO-select-0 .. NCO-select-9) with the bits being active high. Bit 19 : LSB, bit 28: MSB. At the same time bit 29 (NCO-load) is set high. NCO-load will be set low again after 100ns (strobed bit).

NCOPRS Reset the NCO phase to 0.
Hardware action: NCOPRS sets bit 30 (NCO-reset) low. The bit will be set high again after 100ns (strobed bit) which starts the reset procedure (see also subsection 4.2).

RXSYNCON Set the SYNC bit on.
Hardware action: RXSYNCON sets bit 31 (RX-sync) high.

RXSYNCOFF Set the SYNC bit off.
Hardware action: RXSYNCOFF sets bit 31 (RX-sync) low.

RXBITOFF [0] ... [,31] Set RX bit(s) on. There are **no** checks implemented, nor does an interpretation concerning active high/ active low signals take place. Note, use this instruction at your own risk.

eg RXBITON 2,5 set bits 2 and 5 high

RXBITOFF [0] ... [, 31] Set RX bit(s) off. There are **no** checks implemented, nor does an interpretation concerning active high/ active low signals take place. Note, use this instruction at your own risk.

2.5 Sample Program

The following sample shows the structure of a RACELAN program for the transmitter radar controller, there's no intention to show a useful experiment ...

```
% SAMPLE PROGRAM
DEF MAXUNITNO 3           % 4 DDS units are available

AT 1    WREG FSEL0,UNIT0,OPERA
AT 2    WREG FSEL1,UNIT0,OPERB
AT 3    WREG FSEL0,UNIT1,OPERA
AT 4    WREG FSEL1,UNIT1,OPERB
AT 5    WREG FSEL0,UNIT2,OPERA
AT 6    WREG FSEL1,UNIT2,OPERB
AT 7    WREG FSEL0,UNIT3,OPERA
AT 8    WREG FSEL1,UNIT3,OPERB

AT 9    FLOAD UNIT*,OPERA

AT 10   RXPON
AT 11   PREAMPOFF

AT 20   TXSYNCON          % sync pulse for sync control
AT 25   TXSYNCOFF         %

AT 30   TXBITON 4
AT 40   BEAMON

AT 70   MOSEL UNIT0       % use DDS unit 0

AT 79.9 TXBITOFF 3        % sync pulse to check sync with RX on oscilloscope
AT 80   RFDRON
AT 90   TXBITON 3         % note: pulse inverted on RX

AT 440  RFDROFF

AT 450  MOSEL UNIT1       % use DDS unit 1
AT 460  RFDRON
AT 820  RFDROFF


AT 1375 BEAMOFF

AT 1380 TXBITOFF 4

AT 1980 RXPOFF

AT 1993 PREAMPON

AT 6300 CALON      % 200 us calibration pulse at beginning of RX period
AT 6500 CALOFF

SETTCR 0

AT 8000 END
```

3 Instruction Sequences

This section gives an introduction on the sequencing of radar controller instructions, based on the view from the transmitter's side and considering the system's safety. The appropriate timing of instructions for the radar controller (RX) is given in square brackets.

3.1 Transmit Cycle

The following table lists in which sequence the instructions for the radar controller (TX) have to be given to allow transmission.

ANTENNAn
RXPON
PREAMPOFF
set up waveform parameters
[*set up receiver parameters for TX sampling*]
BEAMON
RFDRON
select waveform parameters
[*select receiver parameters for TX sampling*]
ADCTRIGON / ADCTRIGOFF
RFDROFF
BEAMOFF
TXSYNCON, TXSYNCOFF
[RXSYNCON, RXSYNCOFF]

(The sync bits are set independently, the timing for the ADCTRIG bit isn't clear yet.)

Set up/ Select Waveform Parameters Concerning the security of the system, the setup of the following parameters for the waveform can be done in arbitrary sequence. See subsection 4.1 for timing conditions, and the DDS manual for correct DDS setup.

DDS instructions: WREG, FSEL, UNIT, OPER, FLOAD, MOSEL

phase instructions: PHA0, PHA180

3.2 Receive Cycle

This is the sequence for the radar controller (TX) during reception. Basic principle: no transmission during reception. In square brackets: radar controller (RX) receiver timing.

RFDROFF
BEAMOFF
[*set up receiver parameters*]
RXPOFF
PREAMPON
(CALON)
[*select receiver parameters*]
(CALOFF)
PREAMPOFF
RXPON
TXSYNCON, TXSYNCOFF
[RXSYNCON, RXSYNCOFF]

Notes: CALON/ CALOFF is only used when calibrating the receiver chain. The sequence of PREAMPOFF and RXPON is not important.

Set up / Select Receiver Parameters Please refer to the receiver documentation for the correct setup of the parameters. There are no special requirements concerning the security of the system, but refer to subsection 4.2. Available instructions:

ENABMn, DISBMn, BUFFLIP1, BUFFLIP2, SETCOUNT, NCOSELn, NCOPRS, SYMB_NAME

4 Timing Conditions

This section contains the timing conditions caused by the limited hardware's execution time, and also requirements to prevent hardware damage.

4.1 Transmitter Instructions

The following table lists the hardware's execution time. If that time is shorter than a half radar controller cycle it is listed as 0.

instruction	time until status is guaranteed	the instruction must be preceeded by:
BEAMON	$10\mu s$	RXPON, PREAMPOFF
BEAMOFF	$10\mu s$ ($> 5\mu s$)	RFDROFF
CALON	$1\mu s$	PREAMPON (, RXPOFF, BEAMOFF)
CALOFF	$1\mu s$	-
PREAMPON	$\sim 5\mu s$	RXPOFF (, BEAMOFF, RFDROFF)
PREAMPOFF	$\sim 5\mu s$	-
RFDRON	($200ns$)	BEAMON (, RXPON, PREAMPOFF)
RFDROFF	($200ns$)	-
RXPON	$> 10\mu s$	-
RXPOFF	$\sim 10\mu s$	BEAMOFF (, RFDROFF)
WREG	$0.5\mu s$	-
FLOAD	$0.8\mu s$	-
MOSEL	$0.4\mu s$	-
PHA0	(0)	-
PHA180	(0)	-
ADCTRIGON	(0)	-
ADCTRIGOFF	(0)	-
ANTENNA	($1ms$)	-
TXSYNCON	(0)	-
TXSYNCOFF	(0)	-

If there is an instruction listed in column "instruction must be preceeded by" this has to be sent before the one in column "instruction" (the RACELAN compiler checks this.) The table has somehow a recursive structure, because there's often even another instruction to be sent before the one in the last column. Those instructions are listed in round brackets.
Times in round brackets are times estimated by me (ReH) and may become changed. The RACELAN compiler (version 2) checks all above times which thereto need to be listed in the file hardware.dly. Therefore, this file needs to

contain the correct times to prevent possible hardware damage and/or unexpected results.

4.1.1 Power Amplifiers Specification

Special requirements for the power amplifiers timing[3]:

duty factor (RF)	0.1 % .. 25 %
pulse length (RF)	1 μs .. 2 ms
pulse repetition rate (beam)	20 Hz .. 2 kHz
interpulse period (beam)	$\geq$ 0.5 ms

Note: The interpulse period IPP is the reciprocal of the pulse repetition rate. The beam pulse d.c. shall not exceed 30%.

4.1.2 Receiver Protector Specifications

Special requirements for the switching of the receiver protector (please see RXP manual):

pulsewidth	60 μs ... 2050 μs
PRF	0 Hz ... 5000 Hz
duty cycle	$\geq$ 0.3 %

4.2 Receiver Instructions

instruction	time until status is guaranteed	the instruction must be preceeded by:
ENABM		
DISBM		
BUFFLIP1		
BUFFLIP2		
SETCOUNT		
NCOSELn	$0.4\mu s$	
NCOPRS	$< 400ns$	
RXSYNCON	(0)	-
RXSYNCOFF	(0)	-

(End of Document)

[3]see also Harris' manual volume 1, subsection 3, 8928 179 11102, sheet Sh.595-2, Aug/93