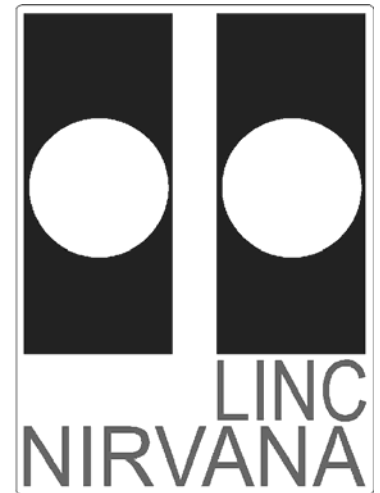


LINC-NIRVANA

The **L**BT **I**Nterferometric **C**amera and
Near-**I**nfra**R**ed / **V**isible **A**daptive
iNterferometer for **A**stronomy

A collaborative project of the MPIA Heidelberg, INAF-Arcetri,
Universität zu Köln, and MPIfR Bonn

<http://www.mpia.de/LINC>



LINC-NIRVANA

-

Science Detector Control Pattern

Doc. No. LN-MPIA-FDR-ELEC-007
Short Title IR Detector Control Pattern
Issue 1.2
Date 27 September 2004

Prepared	<u>Vianak Naranjo</u>	<u>27 September 2004</u>	
	Name	Date	Signature
Approved	<u>Peter Bizenberger</u>	<u>24 January 2005</u>	
	Name	Date	Signature
Released	<u>Martin Kürster</u>	<u>20. May 2005</u>	
	Name	Date	Signature

Document Change Record

Issue	Date	Section/ Paragraph Affected	Reasons / Remarks
0.1	23 July 2004	All	New document
1.0	27 Sept. 2004	5.3 Operation / Idle Loop	Idle break & Idle wait macros
1.1	24 Jan. 2005	All	Approved status
1.2	20. May 2005	All	Released

TABLE OF CONTENTS

Scope	4
1 Applicable documents	4
2 External Interfaces	4
3 Acronyms and abbreviations	4
4 Introduction	4
4.1 General Information	4
4.2 Readout Process Summary	5
4.3 External Processing	6
5 Readout Process of the Detector	6
5.1 Initialization	7
5.1.1 Patterns	7
5.2 Readout Schemes	10
5.2.1 dcr mode : Correlated Double Sampling – frame orientated	10
5.2.2 rrr mode: Correlated Double Sampling with fast reset – frame orientated	11
5.2.3 fcr mode: Correlated Double Sampling – frame orientated	12
5.2.4 lir mode: Line Interlaced Mode.....	13
5.2.5 mer mode: Multiple End-point Read – Fowler sampling.....	14
5.3 Operation.....	15
5.3.1 Cycle Repeat	15
5.3.2 Idle Loop	15

LIST OF TABLES

Table 1. Filenames of the different readout modes and other important information	6
---	---

LIST OF FIGURES

Figure 1. Readout handling	5
Figure 2. Patterns, instruction tables, and macros.....	5
Figure 3. Flow chart of the dcr mode	10
Figure 4. Flow chart of the rrr mode	11
Figure 5. Flow chart of the fcr mode.....	12
Figure 6. Flow chart of the lir mode.....	13
Figure 7. Flow chart of the Fowler mode.....	14
Figure 8. Cycle repeat and Idle loop	15
Figure 9. Idle Break Mode	16
Figure 10. Idle Wait Mode	16
Figure 11. Flow charts for the Idle Wait and Idle Break modes	17

Scope

This document describes the organization of the readout patterns for LINC-NIRVANA's detector, a Rockwell HAWAII-2 PACE detector.

1 Applicable documents

No.	Title	Number & Issue
AD1	Readout Electronics	LN-MPIA-FDR-ELEC-001
AD2	Science Detector	LN-MPIA-FDR-ELEC-006
AD3	Infrared Camera Software	LN-MPIA-FDR-ICS-005

2 External Interfaces

Item	Short description
EI1	Command list of CPB3 firmware (MPIA electronics department)

3 Acronyms and abbreviations

CPB	Control Processor Board
DSP	Digital Signal Processor
GEIRS	MPIA generic infrared detector readout software
HW	Hardware
ICE	Instrument Control Electronics
PATGEN	Pattern Generator Board
PDR	Preliminary Design Review
SW	Software
ROE	Read Out Electronics

4 Introduction

4.1 General Information

The organization of the readout process of a HAWAII-2 detector is based on patterns. These patterns are one of the three units (see Fig.1) used to control the detector. The files define the readout modes, the integration time, the idle function, number of repetitions, etc. The patterns are plain ASCII files, basically the commands accepted by the ROE plus a number of commands interpreted by the GEIRS software for a somewhat convenient control structure. A description of the commands can be found in EI1. The arrangement of the files is designed to manage the complex mechanisms of controlling an infrared detector, as there are various readout modes, idle modes, multiple reads, etc.

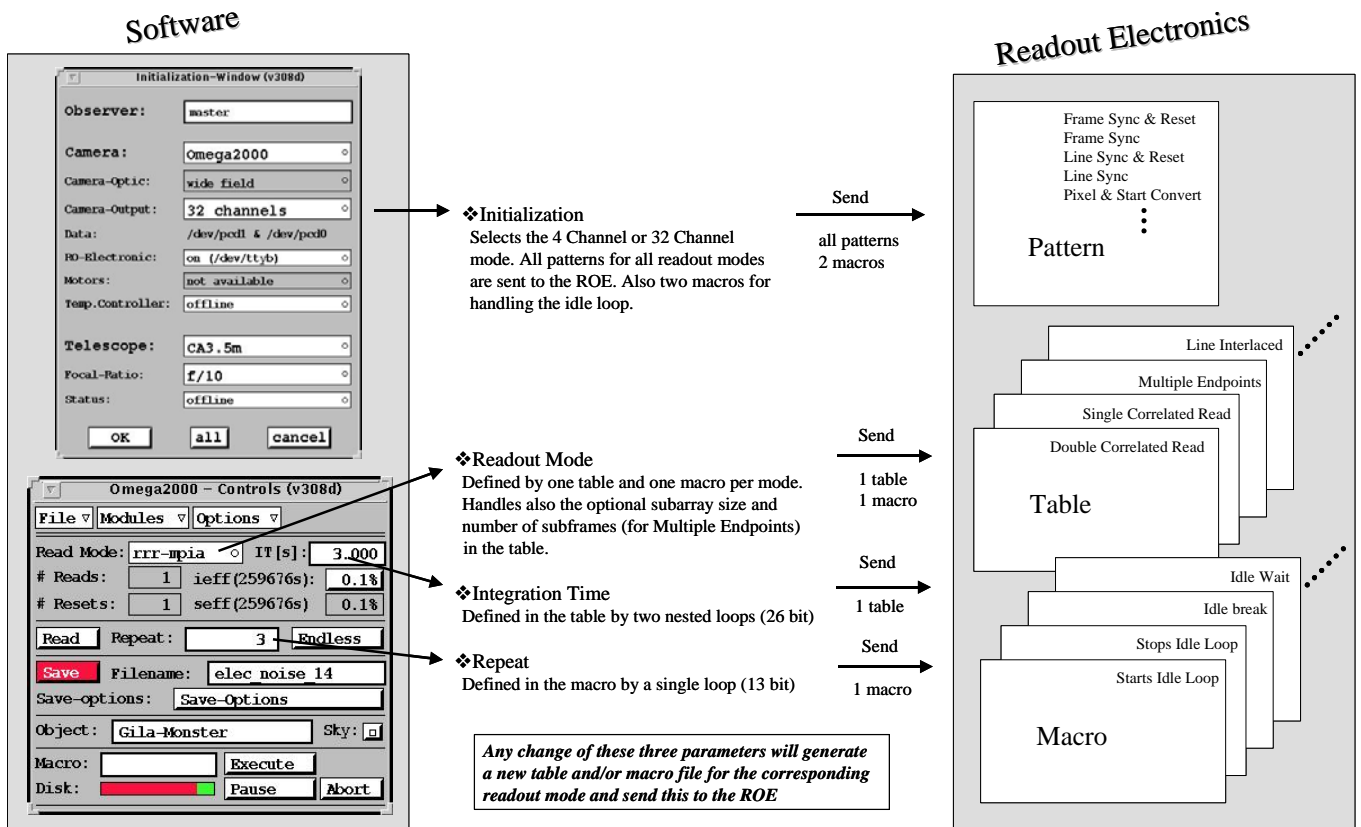


Figure 1. Readout handling

4.2 Readout Process Summary

The detector readout process consists of three main units: patterns, instruction tables, and macros.

The **patterns** are sets of instructions that control the sequence of the signals (max. 48 bit wide) generated by the CPB. Combining these patterns results in a large sequence of signals, which correspond to the different readout modes.

The combination of the patterns is made by the use of **instruction tables**; each readout mode has its own table.

In order for the read out modes to work properly, additional synchronization instructions and specific parameters are needed. This information is stored in **macros**.

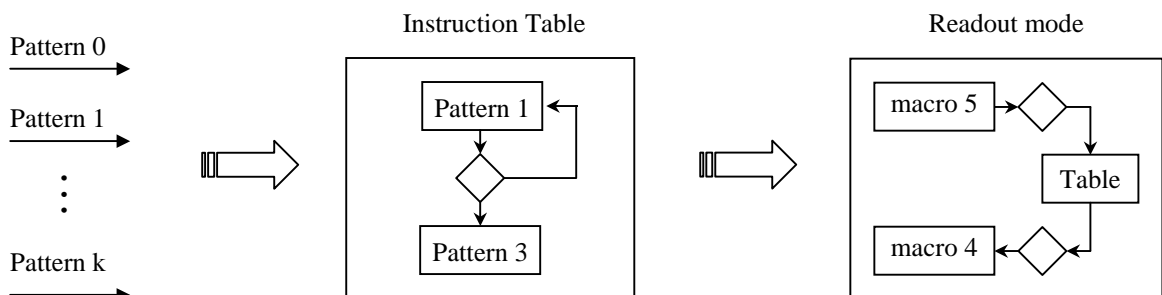


Figure 2. Patterns, instruction tables, and macros

4.3 External Processing

Each read out mode requires a total calculated integration time in order to be processed correctly. All required calculations are done by the GEIRS software. The total integration time is the sum of the frame time during the first read with an additional time before the second read starts (for a double correlated read). This frame time, called *itime_frame*, can be calculated via the shell script *itime_frame.sh*. The value for full frames is constant and is defined and stored in a variables file.

5 Readout Process of the Detector

The required information for the readout process is stored in a series of individual files, which will be described in this chapter. A table of these files with brief descriptions appears below:

Table 1. Filenames of the different readout modes and other important information

File name	Description
roe_init_ch32 roe_init_ch4	Initialization files, including all patterns and macros for the 32 and 4 channel versions
roe_variables	File with the definitions of all the variables used by GEIRS
roe_rdmode_dcr roe_rdmode_rrr-mpia roe_rdmode_fcr roe_rdmode_lir roe_rdmode_mer roe_rdmode_scr roe_rdmode_rr-mpia roe_rdmode_spr	Files for the individual readout modes
roe_crep roe_itime roe_run roe_stop	Defines the number of repetitions Defines the integration time Starts a read Aborts a read
table_dcr table_rrr-mpia table_fcr table_lir table_mer table_scr table_rr-mpia table_spr table_itime	Contain the files for the different readout modes
roe_ptime roe_qreset roe_qreset_ret roe_qvers roe_qvers_ret roe_reset roe_reset_ret roe_status	Files used only by GEIRS for communication with the ROE

5.1 Initialization

All the files used for the readout process consist of ASCII coded commands and parameters accepted and processed by the ROE. More information about the interpretation and description of these can be found in AD1 and EI1.

The ASCII commands included in this document are shown in italics. Parameters beginning with the symbol "\$" are fixed variables that will be replaced in the command lists by the corresponding value according to GEIRS. More information can be found in AD3.

Whether in the 4-Channel (roe_init_ch4) or 32-Channel mode (roe_init_ch32), the ICE and some important system variables of the DSP are reset during the initialization process. The actual version of the DSP Program is sent to the ROE.

105 ... system reset
75 ... send version

After this, the desired PATGEN is also reset, the desired input type for the patterns is chosen, the old pattern files are deleted, and the desired clocks for the output registers and the output drivers for the Signals are enabled.

213 1 ...patgen state machine reset
251 1 ...pattern input hexadecimal
256 1 ...delete all old patterns
211 1 2 3 4 ...enable clocks 2 3 and 4
228 1 3 ...enable output drivers for the Signals (1-16)

The buslink is being set, i.e. the desired control registers are been written.

400 1 1 30 ...setup buslink
400 2 1 30
400 1 1 15
400 2 1 15
400 1 3 1
400 2 3 1
400 1 6 255
400 2 6 255

The Pattern Initialization begins right after this. The memory needed for the different patterns is reserved on the CPB; this patterns will later be loaded to the PATGEN.

5.1.1 Patterns

There are 11 patterns described in the initialization files.

Pattern 0: this is a dummy pattern. It is used for synchronization and frame counting purposes only.

250 1 0 4 1 0 15 ...pattern 0 initialization command
252 C000 0336 ...dummy

Pattern 1: This pattern is the start of a frame. It resets the vertical and horizontal shift registers (FSYNC and LSYNC), sets the vertical register (VCLK) to the first row, and resets all the pixels of the first row.

```
250 1 1 4 14 0 15      ...pattern 1 initialization command
2 252 C000 0336      ...initial state
2 252 C000 0306
2 252 C000 0336
2 252 C000 0376
2 252 C000 0336
2 252 C000 03B6
2 252 C000 0336
```

Pattern 2: this pattern is used to switch to the next line and reset all the pixels in it. LSYNC resets the horizontal register, and VCLK advances the multiplexer one row.

```
250 1 2 4 14 0 15      ...pattern 2 initialization command
2 252 C000 0336
2 252 C000 0316
2 252 C000 0336
2 252 C000 0376
2 252 C000 0336
2 252 C000 03B6
2 252 C000 0336
```

Pattern 3: this pattern is used to go to the first line of the frame. It resets the vertical and horizontal shift registers (FSYNC and LSYNC), and sets the vertical register (VCLK) to the first row. There is no pixel-reset action done here.

```
250 1 3 4 14 0 15      ...pattern 3 initialization command
2 252 C000 0336
2 252 C000 0306
2 252 C000 0336
2 252 C000 0376
2 252 C000 0336
2 252 C000 0336
2 252 C000 0336
```

Pattern 4: this pattern switches to the next line without resetting the pixels in it. LSYNC resets the horizontal register, and VCLK advances the multiplexer one row.

```
250 1 4 4 14 0 15      ...pattern 4 initialization command
2 252 C000 0336
2 252 C000 0316
2 252 C000 0336
2 252 C000 0376
2 252 C000 0336
2 252 C000 0336
2 252 C000 0336
```

Pattern 5: this patterns clocks the pixels and starts data conversion.

```
250 1 5 4 72 0 15          ...pattern 5 initialization command
22 252 C000 0339
1 252 0000 0339
2 252 C000 0339
47 252 C000 0336
```

Pattern 6: sets the integration time for pattern 1.

```
250 1 6 4 $itime_lines1 0 15      ...pattern 6 initialization command
$itime_lines1 252 C000 0236
```

Pattern 7: pixel clock without starting data conversion (for first pixel).

```
250 1 7 4 72 0 15          ...pattern 7 initialization command
25 252 C000 0339
47 252 C000 0336
```

Pattern 8: starts the data conversion without pixel clock.

```
250 1 8 4 36 0 15          ...pattern 8 initialization command
22 252 C000 0336
1 252 0000 0336
5 252 C000 0336
8 252 C000 0336
```

Pattern 9: sets integration time for pattern 2.

```
250 1 9 4 $itime_lines2 0 15      ...pattern 9 initialization command
$itime_lines2 252 C000 0236
```

Pattern 10: resets the horizontal shift register (LSYNC) and all the pixels. No clocking of the vertical shift registers.

```
250 1 10 4 14 0 15         ...pattern 10 initialization command
2 252 C000 0336
2 252 C000 0316
2 252 C000 0336
2 252 C000 0336
2 252 C000 0336
2 252 C000 03B6
2 252 C000 0336
```

Once all the patterns are defined, the initialization process continues with the definition of macros. The description of each macro can be found in the Operation section later in this document.

5.2 Readout Schemes

In order to run the different readout modes, the patterns have to be properly arranged. This arrangement is done by means of instruction tables, which are described below.

5.2.1 dcr mode : Correlated Double Sampling – frame orientated

In this mode all lines are reset in succession, and then two frames are read in succession. This allows a longer period between the resetting of a pixel and the first read of this pixel.

The instruction table of this mode (*table_dcr*) can be represented by the following flow chart:

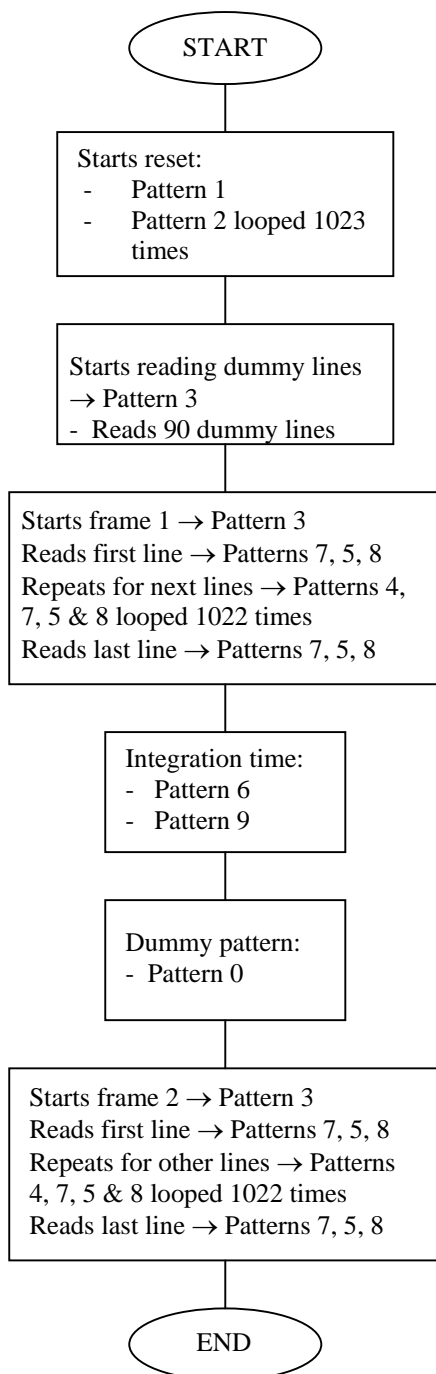


Figure 3. Flow chart of the dcr mode

5.2.2 rrr mode: Correlated Double Sampling with fast reset – frame orientated

This mode makes use of the fast reset of a complete line. Immediately after the reset of one line, the same line is read. After applying this to all lines, all lines are read again (without a reset). In this scheme there is basically no overhead for resetting the detector, since the reset is fast compared to reading one line. This readout mode for HAWAII detectors was developed at MPIA.

The following flow chart illustrates the process described in its instruction table (*table_rrr-mpia*).

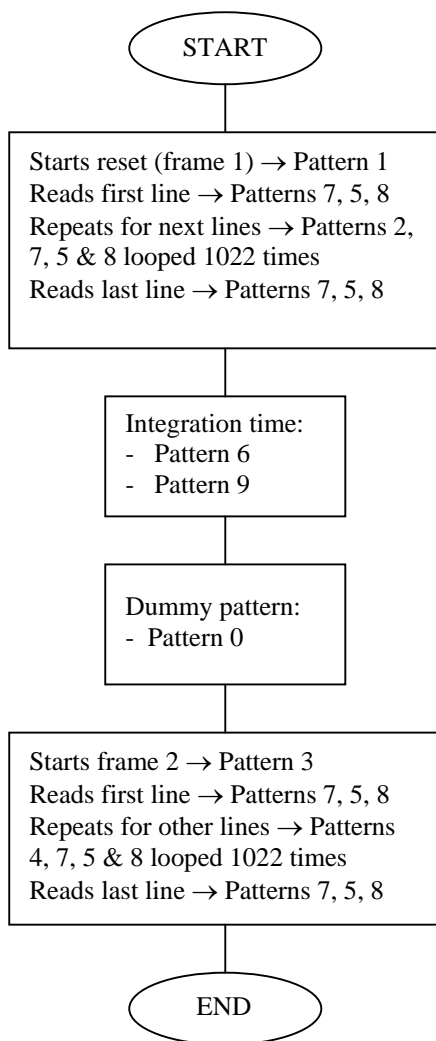


Figure 4. Flow chart of the rrr mode

5.2.3 fcr mode: Correlated Double Sampling – frame orientated

In this mode, after the reset of the first line, the same line is clocked as if it would be read. However, no data conversion occurs. This is applied to all lines. Then, the two frames are read in succession. The flow chart of this process (*table_fcr*) is presented below.

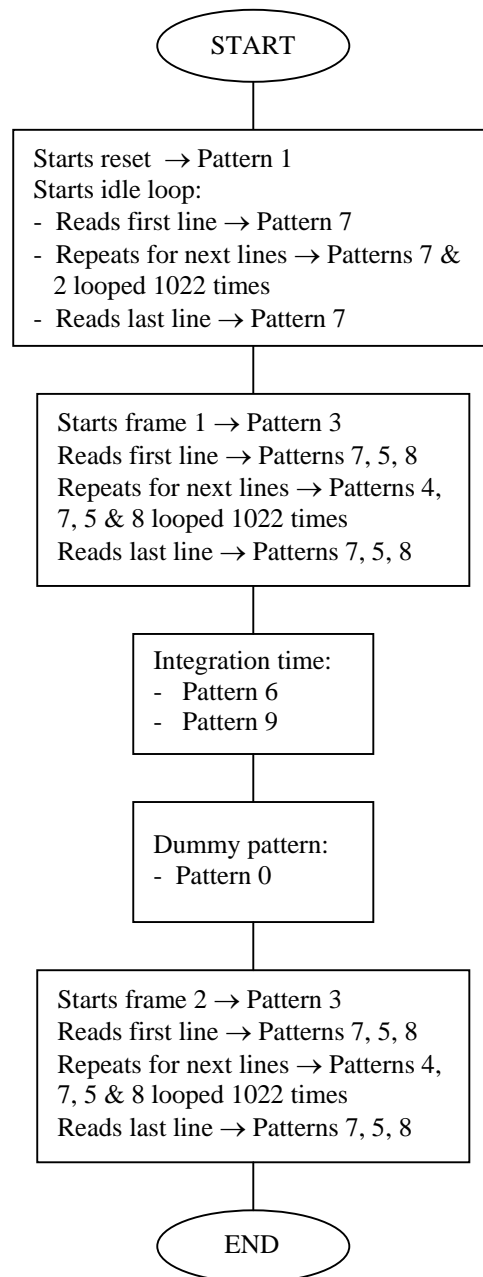


Figure 5. Flow chart of the fcr mode

5.2.4 lir mode: Line Interlaced Mode

In this mode, each line of the detector is read, then reset, and then read again immediately. The flow chart of the instruction table of this mode (*table_lir*) is presented below.

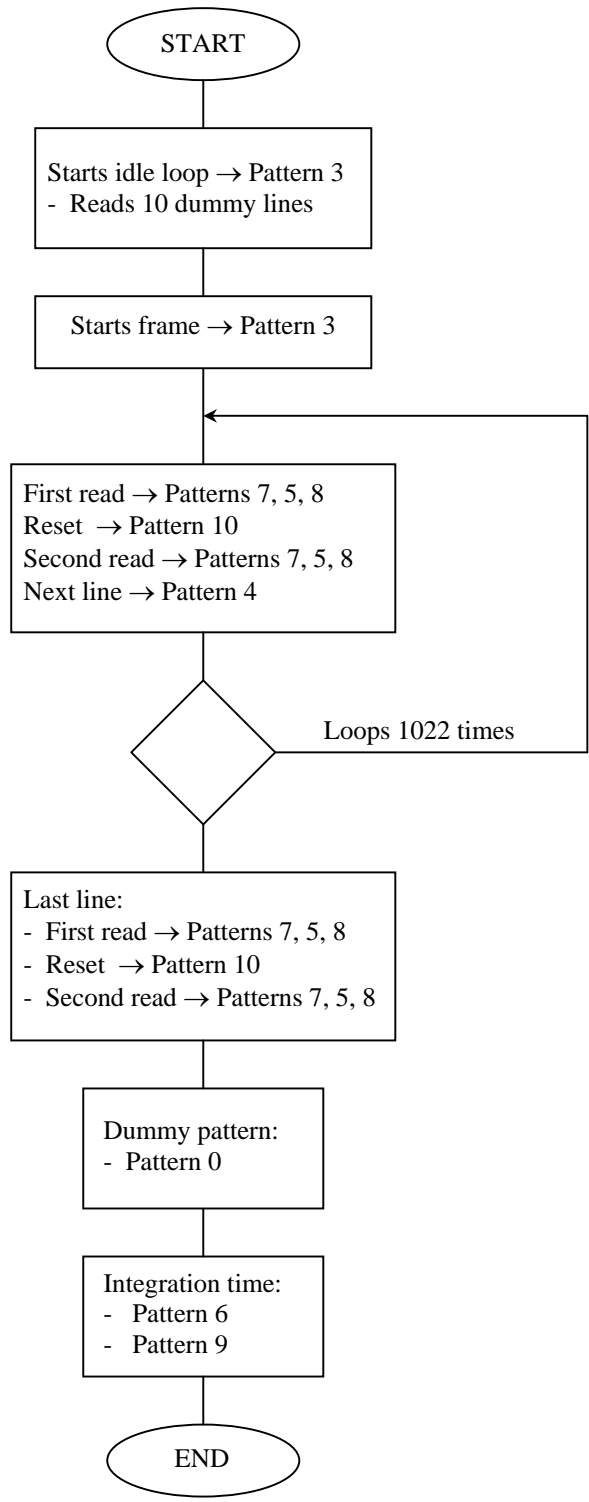


Figure 6. Flow chart of the lir mode

5.2.5 mer mode: Multiple End-point Read – Fowler sampling

This mode is still not yet fully implemented. The strategy is to reset the whole frame, read it several times, and after the integration time, read it the same number of times again. This reduces the effective read noise.

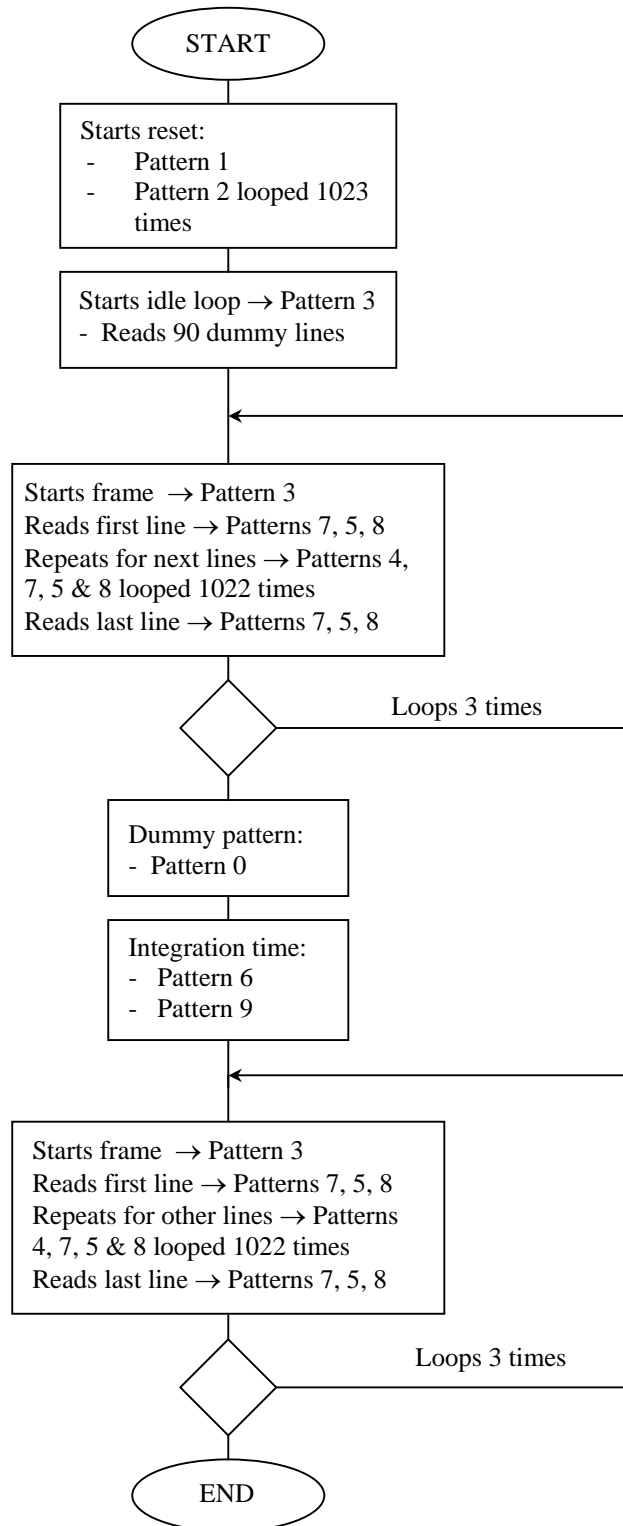


Figure 7. Flow chart of the Fowler mode

The flow charts and descriptions above correspond to the modes most used for science. The rest of the modes listed in Table 1 are for engineering purposes only.

5.3 Operation

5.3.1 Cycle Repeat

A cycle is defined as the process in which reading, integration (Δt_i), and data conversion takes place. The Cycle Repeat (crep) is just the way to tell the system how many times a cycle is going to be repeated, and it is defined in a macro by a single loop (13 bit).

5.3.2 Idle Loop

The Idle Loop occurs after the cycle repeat, and is the process by which reading and integration (Δt_i) continues to take place, but there is no data conversion, i.e. the ADC's are not enabled.

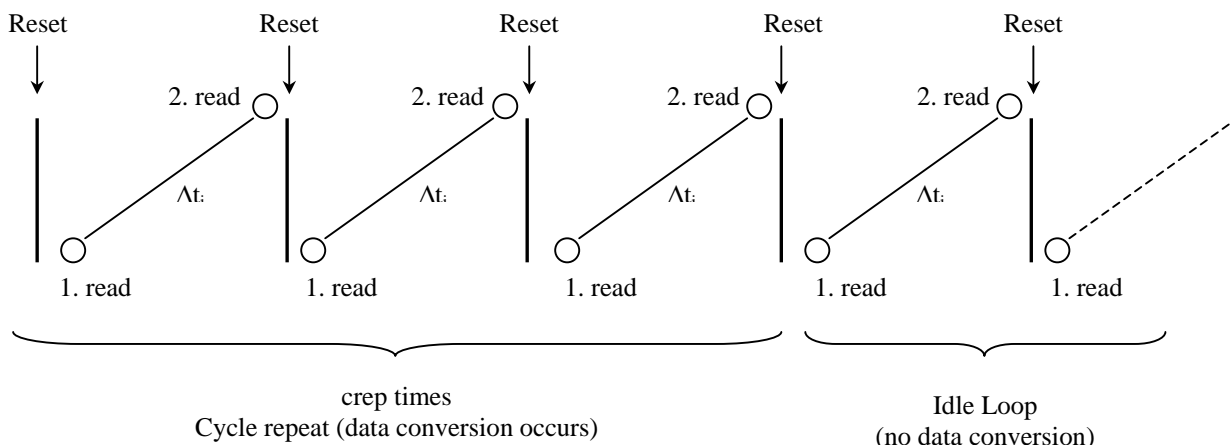


Figure 8. Cycle repeat and Idle loop

The **roe_crep** file contains two macros which show that there are 2 ways for handling the idle loop: idle break and idle wait.

In the Idle Break Mode (macro #12 – see below), the process is interrupted during the integration in order to immediately begin with the next data conversion. On the other hand, the Idle Wait Mode (macro #11 – see below) waits until the integration finishes before beginning with the next data conversion, i.e. the next cycle repeat.

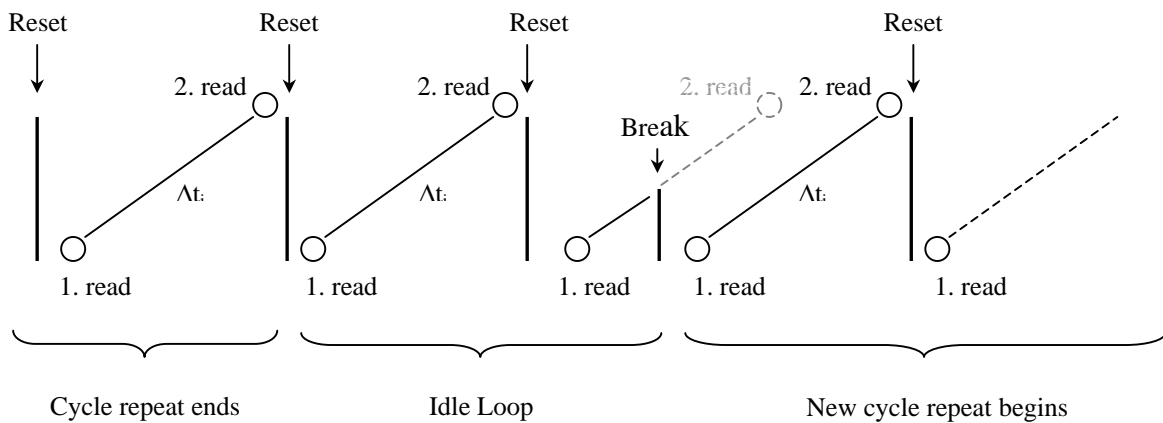


Figure 9. Idle Break Mode

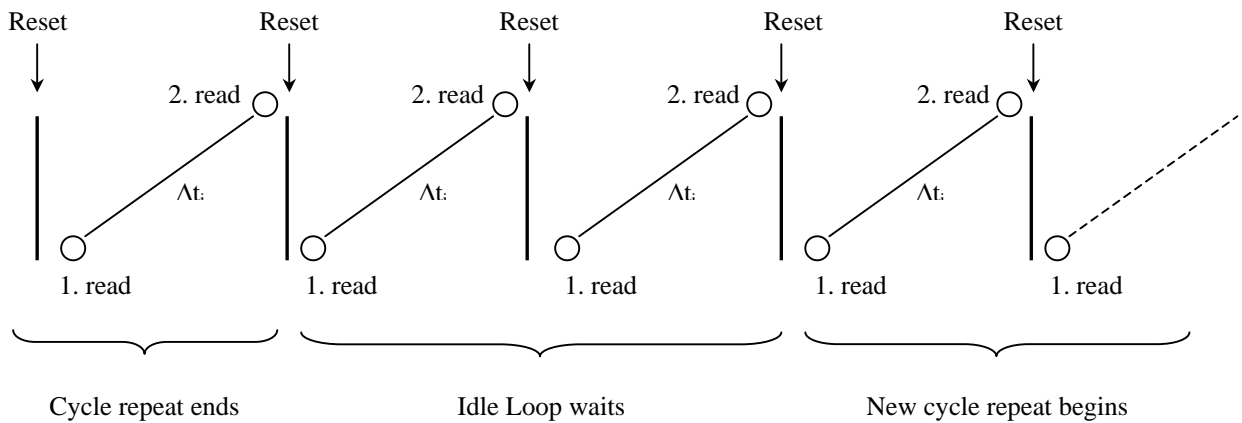


Figure 10. Idle Wait Mode

The ASCII commands and parameters of the macros corresponding to these idle modes, as well as its flow charts are shown below:

```

505 11
506 220 1 3 0X40
506 232 1 1
506 211 1 2
508 0 7
506 753 1 4 501 11
509
508 $crep 5
506 232 1 1
506 212 1 2
...macro#11: starts the idle wait
...setup sync source to EOE
...turn on EOE sync
...enable the start conversion clocks
...jump to line 7
...wait for EOE, then continue
...break
...jump to line 5, loop "crep" times
...enable EOE sync
...start conversion off
    
```

505 12 ...**macro#12**: starts idle break
 506 220 1 3 0X80 ...tricky, must not be 0x40, anything else is ok
 506 232 1 0 ...disable EOE sync
 506 213 1 ...PATGEN state machine reset
 506 231 1 1 ...endless bit on (continuous scan on)
 506 211 1 2 ...start conversion on
 506 210 1 ...start PATGEN
 508 1 10 ...jump to line
 506 753 1 8 501 12 ...wait for End of pattern 0, then continue
 509 ...break
 508 \$crep 8 ...jump to line 8, loop “crep” times
 506 220 1 3 0X40 ...setup sync source to EOE
 506 232 1 1 ...enable EOE sync
 506 212 1 2 ...start conversion off

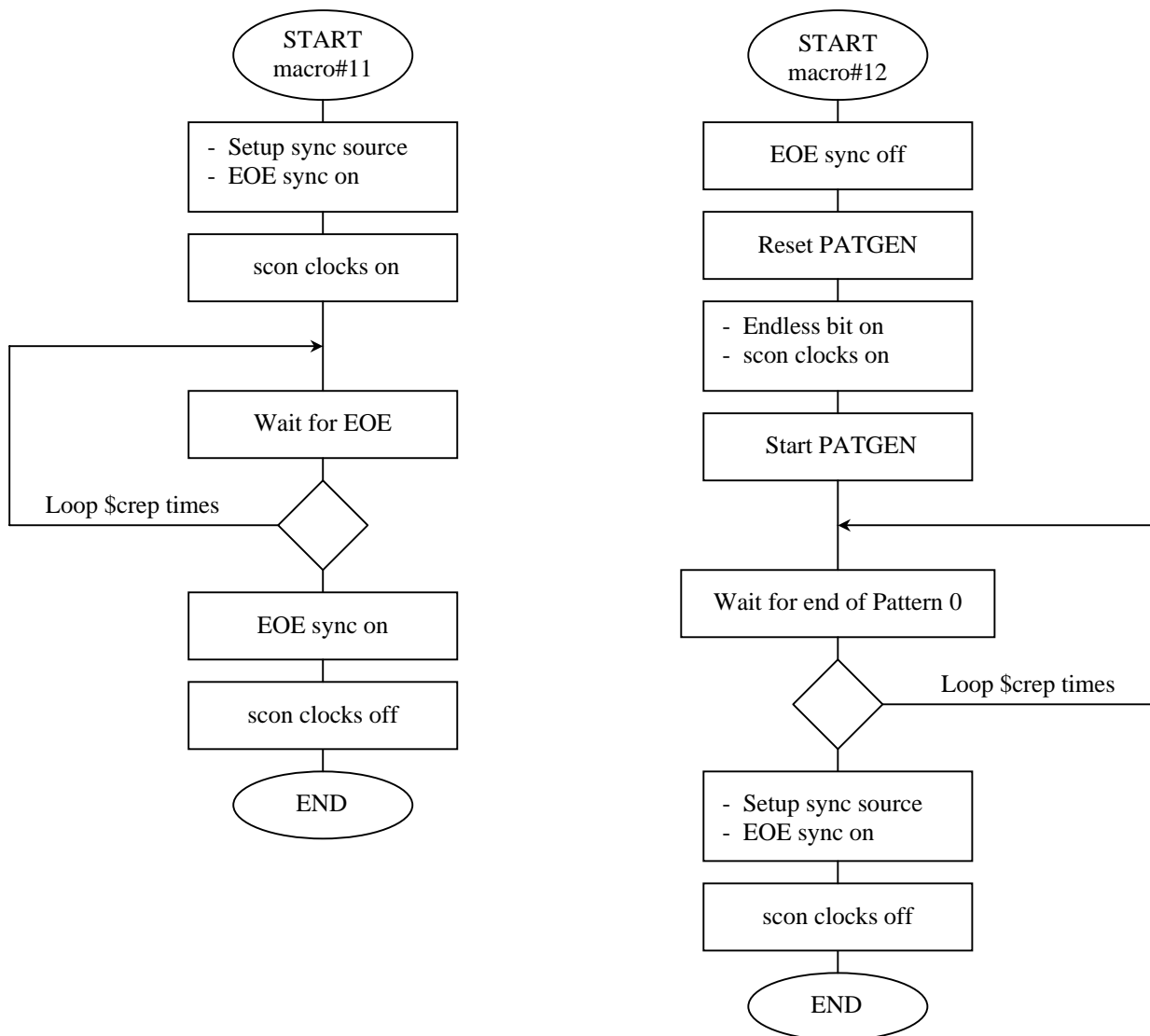


Figure 11. Flow charts for the Idle Wait and Idle Break modes

Since the whole operation of the read out process is made by means of macros, there are still two important macros to mention. They are defined during the initialization process, and are in charge of the starting and stopping of the idle loop:

```

505 4          ... macro #4: starts the idle loop
506 276 1 6   ...send table to PATGEN
506 231 1 1   ...endless bit on (continuous scan on)
506 212 1 2   ...start conversion off
506 210 1     ...start PATGEN

505 5          ...macro #5: stops the idle loop
506 213 1     ...PATGEN state machine reset
506 212 1 2   ...start conversion off

```

The other important files to mention are the ones that run the different readout modes. Its structure is always the same, they differ only on the table been called. For example, the **roe_rdmode_lir**:

```

500 5          ...start macro #5
263 1 6       ...delete old table
include table_lir ...include desired table; here for example table_lir
213 1         ...stop PATGEN
276 1 6       ...send desired table (here table_lir) to pattern generator
500 4         ...start macro #4

```

The last file to describe is the **roe_itime** file, which defines the integration time (Δt_i) for each readout mode.

```

500 5          ...start macro #5
263 1 6       ...delete old table
include table_$ctype ...table_$ctype corresponds to the desired readout mode table
213 1         ...stop PATGEN
276 1 6       ...send table_$ctype to pattern generator
500 4         ...start macro #4

```

---oOo---