SCIPP 2000/03
January 2000

Setup for Functionality Test of GLAST Front-end Electronics GTFE64, GTRC, and HDI

Masaharu Hirayama

Santa Cruz Institute for Particle Physics
University of California, Santa Cruz
Santa Cruz, CA 95064

Overview

This document describes the setup for a functionality test of the GLAST tracker front-end electronics used for a beam-test tracker built in 1999. With the test setup, a diced front-end chip (GTFE64), a diced controller chip (GTRC), a wafer of either chip, and a high-density interface (HDI) can be tested. You need to be familiar with GTFE64, GTRC, and HDI to understand this document.

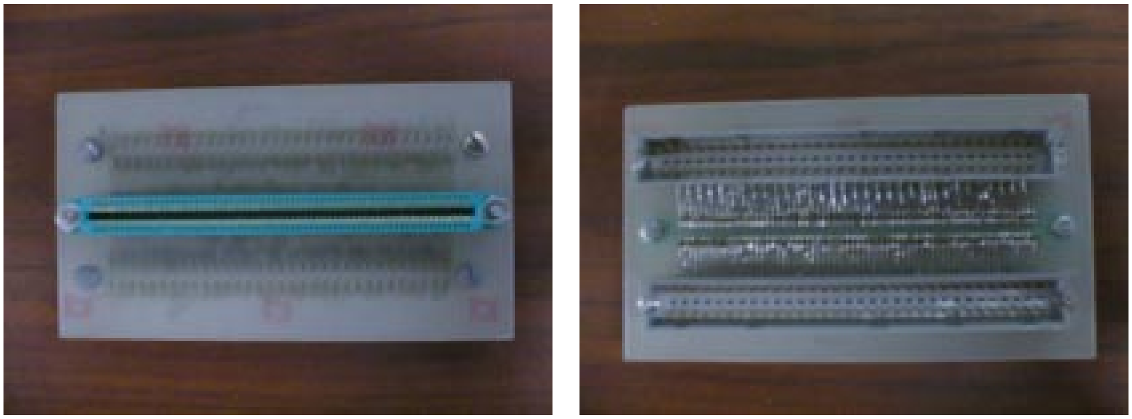
1 Required Instruments

□ *Probe Station, Probe Card, and Probe Card Adopter*

For a test of diced-chip and a wafer, you need a probe station and a special probe card for it. You can use any probe station, automatic or manual, for a diced-chip test. An automatic probe station is required only if you want to test a wafer fully automatically. There are two different types of probe card: the one is for GTFE64c and the other for GTRC. Both are made by a company called "Rucker & Kolls". Passive components are mounted on both cards. A probe card for GTFE64c has probe tips for all pads except for analog-input pads and calibration-input pads. A probe card for GTRC has probe tips for all pads on a chip. Keywords for finding those probe cards are given in a table below.

To probe	DEV. #	R & K #	STAMP #
GTFE64c	GTFE64C	RK19529C	GTFE64C
GTRC	GTRC	RK19564C	GTRC

At the edge of a probe card, a probe card adopter is attached for electrical connections. One probe card adopter can be used to test both GTFE64 and GTRC.



Probe card adopter (left) probe card side (right) interface board side

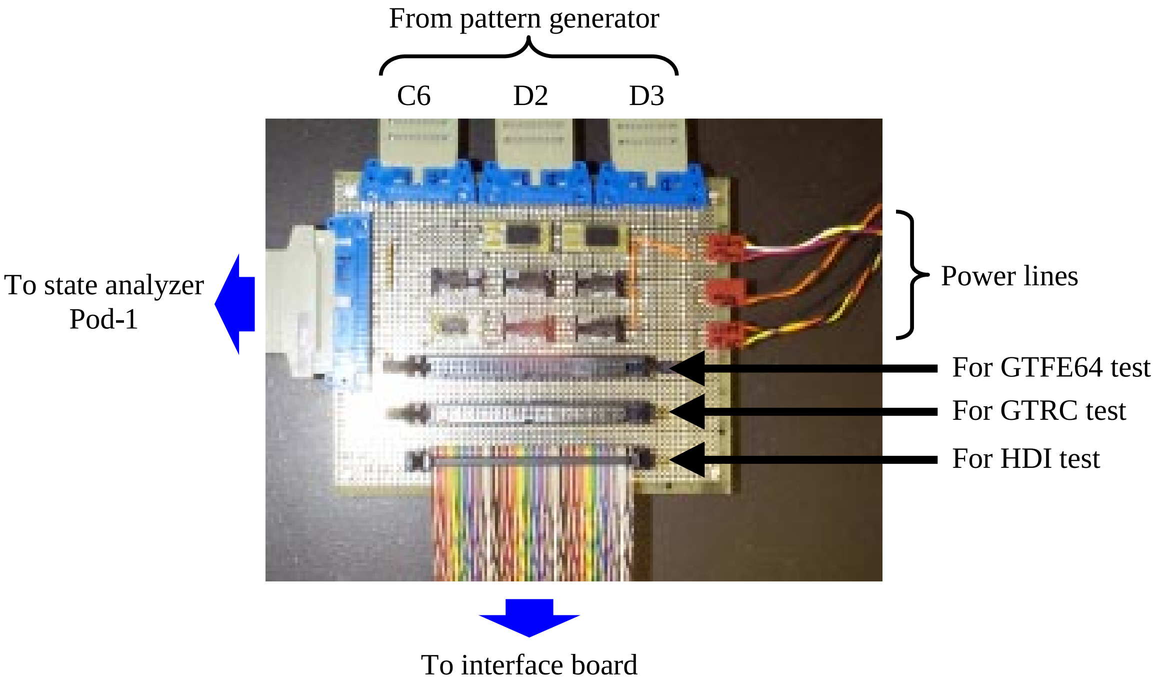
□ *Logic Analysis System HP16500*

The Logic Analysis system HP16500 is used to produce digital signals fed to GTFE64, GTRC, or HDI and to sample and record its response. For those purposes, the system must include a pattern generator module HP16520 and a state analyzer module HP16550 or HP16510. For testing GTFE64 and GTRC, at least one extension card HP16521 is needed for the pattern generator, since more signal lines are needed. In the test, the Logic Analysis system is fully controlled and readout by a computer program through GPIB.

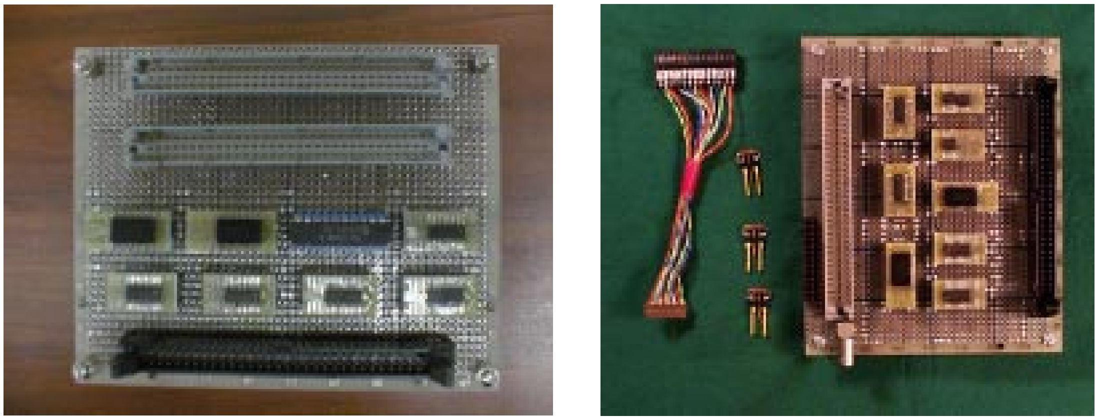
□ *Converter Board and Interface Boards*

Signals from and to the Logic Analysis system are a TTL signal and those from and to GTFE64, GTRC, and HDI are an LVDS signal or a low-voltage CMOS signal (single-ended). Also, those chips are not designed to drive a long cable, such as a meter-long cable. To adopt signal levels and to buffer output signals from a chip being tested, a converter board and an interface board is inserted between the Logic Analysis system and a probe card.

The converter board, connected to the Logic Analysis system, converts a TTL signal from/to the Logic Analysis system to/from an LVDS signal travelling along a long cable to the interface board. You may put the converter board close to the Logic Analysis system. The interface board, connected to a probe card, converts an LVDS signal from/to the converter board to/from low-voltage CMOS signal if necessary. All the LVDS signals from a chip being tested are buffered on the interface board. LVDS signals from the converter board are not buffered and go through the interface board.



The converter board fully connected to the Logic Analysis system (cables on the top and the left side), the interface board (bottom), and the power supplies (right).



(Left panel) Interface board for GTFE64. (Right) Interface board for GTRC and HDI. A special cable and 100 Ω terminators for HDI test are also shown in the picture.

- ***Power Supplies and High Voltage supply***

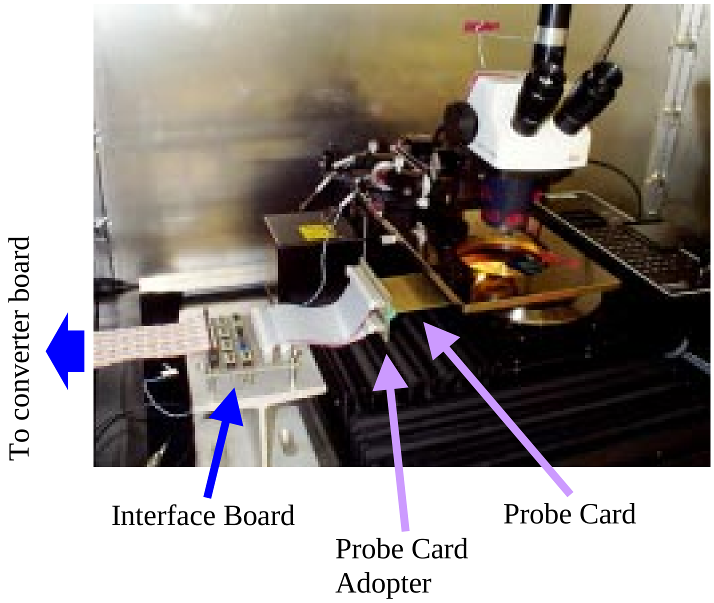
To power electronics to be tested (GTFE64, GTRC, or HDI) and a converter board and an interface board, a couple of voltages should be supplied. The voltages needed are listed in a table below. For a test of HDI with detectors electrically connected, you also need a high voltage power supply to bias the detectors.

Power	Voltage	Remark
DVDD	+3V(*)	
AVDD	+5V(*)	Not needed for GTRC test
AVDD2	+2V	Not needed for GTRC test
+3V	+3V(*)	For converter board and interface board
+5V	+5V(*)	Same as above

(*) To measure current on DVDD, it is recommended to supply DVDD and +3V (for converter board and interface board) separately. To measure current on AVDD, it is also recommended to supply AVDD and +5V (for converter board and interface board) separately.

2 How to Setup

To test a GTFE64c chip or a GTRC chip, set a probe card for them on a probe station. You can use any probe station except the case of testing a wafer, in which you need an automatic probe station. Then carefully attach the probe card adopter at the edge of the card, since the card edge is fragile. Make sure the adopter is plugged in to the bottom of the connector. Place the interface board right next to it, and connect a 64-pin DIN connector on it to one on the probe card adopter with short flat cables. You need two 64-line straight cables (i.e., pin 1 at one end is connected to pin 1 at the other end) for GTFE64c test to connect DIN connector "A" on the interface board to "A" of the probe card adopter, and "B" to "B". For GTRC test you need only one cable to connect "B" to "B". The cables should be as short as possible to reduce unnecessary load to a chip being tested.



Take three cables from the pattern generator, D3, D2, and C6, and plug them in to the converter board as shown in the picture of the converter board in the previous section. Also, plug in the Pod-1 cable from the state analyzer as shown, and connect power lines as shown. Bring 64-line flat cable (straight) and connect the converter board to the interface board with the cable. The connector on the converter board to connect depends on what to test. Choose the right one as indicated in the picture in the previous section. This cable carries only LVDS signals (and power lines), so you can make this cable long enough to place the converter board away from the probe station.

To test an HDI, you setup similarly, but without a probe card, a probe card adopter, or flat cables to connect the interface board to the probe card. Instead, you need a cable to connect an HDI to a 26-pin two-row header on the interface board, which is located at the next to the 64-pin DIN connector. Bring the cable and connect the interface board to an HDI to be tested.

To test a wafer, a test program automatically measures currents on the power lines (DVDD, AVDD, and AVDD2) and records it. For that purpose, power supplies should be HP662x (HP6629A prefered for a GTFE64c test) and Keithley 237 and the power lines should be connected as in the table below.

Power	Voltage	Instrument to connect
DVDD	+3V(*)	Keithley 237
AVDD	+5V(*)	HP662x channel 1
AVDD2	+2V	HP662x channel 2
+3V	+3V(*)	HP662x channel 3 or 4
+5V	+5V(*)	HP662x channel 3 or 4

3 How to Test Electronics

An electronics test with the system described here is basically a functionality test, rather than a measurement of characteristics of the electronics such as noise scan. The system tests a chip or an HDI through digital inputs and outputs only. Therefore, no channel inputs of GTFE64c are tested with this system, while digital functionality of a chip is fully tested. Analog functionality in a GTFE64c chip is tested with a strobe command, which injects charge into specific channels of a chip. Also, no noise measurement should be included in a test with this system, since the setup is very noisy to measure it. The noise comes from the logic analysis system, a probe-station controller, or environmental interference picked up by exposed traces on a probe card.

A test of a single chip consists of three parts. First, the test system sends a set of bit sequences (“test pattern”) to a target electronics (a chip or an HDI). Then, it records target’s response, or a set of bit sequences coming out on digital outputs of the target (“output pattern”). And finally, it compares the response with the one which the chip is supposed to output (“template pattern”). If the comparison results in no difference between two files, the chip passes the test. Otherwise, the chip has some problems to be diagnosed by an expert.

A test pattern and a template pattern should be prepared in pair. An output pattern will be stored in a text file by a test program described below. Result of comparison of an output pattern with a template pattern will also be output in a text file. Both of those test files are important for an expert to diagnose the problems. A tester should keeps all those files for an expert even when he or she repeats a test on a target.

3.1 Test Programs

There are two LabVIEW VI’s to test electronics: electronicsTest.vi and waferTest.vi. The former is for a manual test of individual chips and the latter for an automatic test of a wafer. However, these VI’s work similarly otherwise, since they use the same set of C++ functions to communicate with the HP logic analysis system through a GPIB cable. To test a single chip or a single die, it sends a test pattern to the pattern generator, arms the state analyzer, runs the pattern, and reads out bit streams recorded in the state analyzer.

They also share input file formats for a test pattern and a template pattern, and an output file format for an output pattern. Both of them output a log file which includes results from a test. The four files are all an ASCII file, called a command file, an output file, a template file, and a log file in the test programs. A command file includes commands for the test programs to create a test pattern to be sent to the pattern generator. Its format will be described in a section below. A template file includes a template pattern in a format also described in another section below. An output pattern will be written into an output file, and results from its comparison with a template pattern into a log file.

For a wafer test, waferTest.vi moves the chuck to travel over a wafer and logs test results and currents on power lines. The motion is fully automatic in automatic mode, in which there is no need for human assistance during a test of entire wafer, and is semi-automatic in manual mode, in which the VI tests a wafer die by die. Also, you can switch between those modes at any time of your test. So, you can start testing in automatic mode after you test a few dice in manual mode at first to make sure the entire system is working.

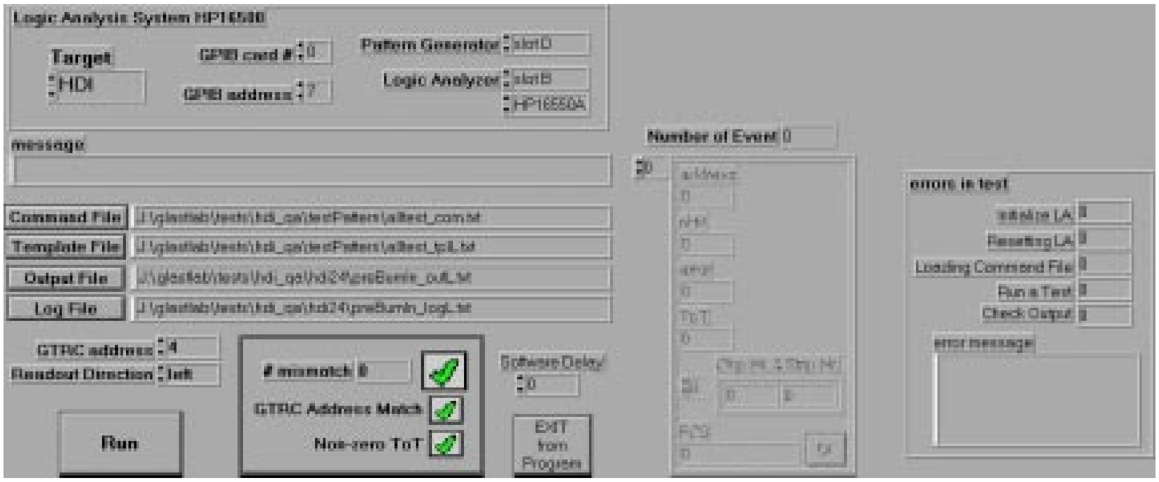
3.1.1 How to use electronicsTest.vi

Before running the VI, you should set configuration of the setup into a box at the top. Choose a target of a test (GTFE64, GTRC, or HDI). Type in the GPIB card number connecting to the logic analysis system and GPIB address of it. Choose a slot of (a master module of) the pattern generator installed in it (slot A through E), a slot of the state analyzer (slot A through E), and a type of the state analyzer (HP16550A or HP16510A).

After setting them correctly, run the VI. The VI initializes the logic analysis system for you in a minute or so, while the message window shows you that it is initializing it. If initialization is failed, check the setting in the top box, exit from the program by pushing the "Exit from the Program" button, and restart the VI with correct setting.

Once the logic analysis system is initialized, specify file names of a command file, a template file, an output file, and a log file. The file names currently set are shown in a box at the middle of the left half. To change a command file name, for example, click the "Command File" button. A window will pop up for you to find the right one for your test. Click a corresponding button to change other file name. You should change an output file name and a log file name every chip. If you choose an existing file for either of them, the VI will ask you whether you want to replace it with a new one or cancel a test. If you answer "to replace it", then the VI overwrites new outputs onto the existing file you specify.

For an HDI test, type in GTRC address and choose Readout Direction (left or right) of your test in a box at the bottom left. For other tests, you don't have to fill those fields.



Logic Analysis System HP16500

Target: HDI
GPIB card #: 0
GPIB address: 7
Pattern Generator: slotD
Logic Analyzer: slotB
HP16550A

message:

Command File: J:\glastlab\tests\hdi_qa\testPattern\alltest_com.txt
Template File: J:\glastlab\tests\hdi_qa\testPattern\alltest_tpl.txt
Output File: J:\glastlab\tests\hdi_qa\hdi24\preBurnin_out.txt
Log File: J:\glastlab\tests\hdi_qa\hdi24\preBurnin_log.txt

GTRC address: 4
Readout Direction: left

mismatch: 0 ✓
GTRC Address Match ✓
Non-zero ToT ✓

Software Delay: 0

Number of Event: 0

address: 0
nHit: 0
error: 0
ToT: 0
Chip Hit & Strip Hit: 0 0
FCS: 0

errors in test
Initialize LA: 0
Resetting LA: 0
Loading Command File: 0
Run a Test: 0
Check Output: 0

error message:

Front panel of electronicsTest.vi. The left half includes test settings, major control buttons, displays of test results, GPIB settings, and so on. The right half does an event display and error messages of the test.

Now you are ready to run a test. For a diced-chip test, load a chip, align it and a probe card, and touch the chip with the probe card. Make sure all the probe tips make a good contact to a appropriate probe pad on the chip. In case of an HDI test, connect an HDI to the test system. Push the “Run” button at the bottom left corner and wait for a message “Done” in the message window. Once a test is done, test results are displayed and stored in an output file and a log file. The VI shows the number of mismatches between an output pattern and a template pattern. The number must be zero to pass a test. If it is zero, a green check mark appears in the box next to the number of mismatches (as in the picture of the VI front panel). If not, a red cross-mark appears in the box and you should ask an expert to diagnose it. For diagnosis, an output file and a log file are important. If you repeat a test, keep all the output and log files separately for diagnosis.

For an HDI test, the VI also checks GTRC address and time-over-threshold. Results are also shown in a box with a green check or a red cross-mark. An HDI should give three green check marks to pass a test. If it does not pass a test, ask an expert to diagnose it. Again, note that an output file and a log file are important for diagnosis. At the bottom of the picture of the VI front panel, there are three green check marks, which means the HDI has passed the test.

3.1.2 How to use waferTest.vi

The VI waferTest.vi requires more setups than electronicsTest.vi. It uses NetDDE to communicate with LabMaster, which controls the automatic probe station. So, you should setup NetDDE on a computer which runs LabMaster before starting waferTest.vi. A power supply HP662x and Keithley 237 is also required to measure currents on power lines (DVDD, AVDD, and AVDD2) automatically. And waferTest.vi works only with the state analyzer HP16550A for a historical reason.

Once NetDDE is setup, invoke LabMaster and load a wafer map. A wafer map should contain a die comment for each die on a wafer to be tested, because a die comment will be used to generate output file names for each die automatically. Load a wafer, align it and a probe card, and set a home position on a home die. Make sure a probe card makes good electrical contacts all over the wafer. Turn on the power supplies and let them output voltages.

Before running the VI, you should set configuration of the setup into a box at the top left corner. Choose a target of a test (GTFE64 or GTRC) and type in a file name prefix, which will be used to generate output file names for each die. At the top center, type in the GPIB card number connecting to the logic analysis system and GPIB address of it. Choose a slot of (a master module of) the pattern generator installed in it (slot A through E) and a slot of the state analyzer (slot A through E). At the middle center, type in GPIB addresses for HP662x and Keithley 237.

Set tolerance on the number of bad dice, if you plan to run a test in an automatic mode. In an automatic mode, the VI stops testing if it finds this number of bad dice straight in a row. This gives you a chance to diagnose the testing system itself when something goes wrong.

Before starting the VI, you have to specify file names of a command file and a template file, and name a directory for output files and log files to be stored. Boxes to type in are at the bottom center. The VI uses the command file and the template file throughout the entire wafer and outputs an output file and a log file for each die to the directory which you specify. File names of an output file and a log file for each die will automatically be generated from the prefix which you specify (see above) and a die comment (see table below for formats).

File	File name
Output file	{prefix}{die-comment}_{date-time}_out.txt
Log file	{prefix}{die-comment}_{date-time}_log.txt
Summary file	summary_ {date-time}.txt

(NB) {date-time} is in format of YYYYMMDD_hhmm where YYYY, MM, DD, hh, and mm are year, month, date, hour, and minute of the system clock when the VI starts.

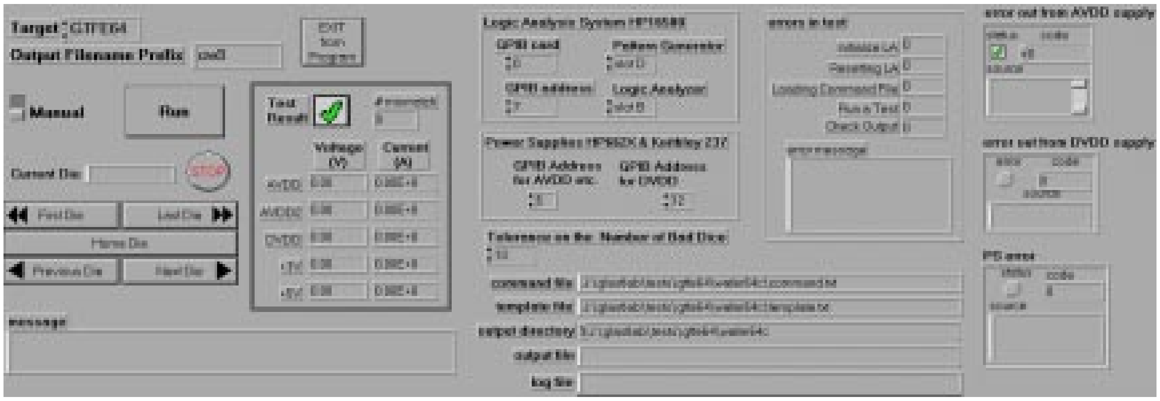
After setting all of above correctly, run the VI. The VI initializes the logic analysis system for you in a minute or so, while the message window shows you that it is initializing it. If initialization is failed, check the setting in the top box, exit from the program by pushing the “Exit from the Program” button, and restart the VI with correct setting.

Also, the VI creates a “summary file” and start logging all the tests you perform from then on until you exit from the VI or stop the VI. A summary file is an ASCII file in a format of a column-wise table. Each line of a summary file corresponds to each die you have tested. It includes, a die comment, currents on power lines (DVDD, AVDD, and AVDD2), a test result which is either of “OK” or “NG” followed by the number of differences between an output pattern and a template pattern in parenthesis, and an output file name.

Once the logic analysis system is initialized, you are ready to run a test. Move to a die to be tested using buttons at the bottom left, indicated as “First Die”, “Last Die”, “Home Die”, “Previous Die”, or “Next Die”, and make good electrical contacts on all the pads to be probed. Choose a run mode (manual or automatic) and push the “Run” button at the middle left. A manual mode is strongly recommended at first. A die comment of the die currently tested will be shown in “Current Die” box below the “Run” button.

In a manual mode, the VI runs a test and pauses after showing test result in a box next to “Run” button. A test result is shown as the number of mismatches between an output pattern and a template pattern. If it is zero, the die has passed the test and a green check mark will indicate that it has (as shown in a picture of a front panel). If not, the die needs a diagnosis by an expert and a red cross-mark will appear.

In automatic mode, the VI runs a test, showing the result, moves to the next die, and repeats them until it reaches the last die. The order of testing is determined by LabMaster, so you can’t change it for automatic testing. During an automatic test, you can stop the run by pushing “STOP” button just below “Run” button; the VI will pause after the current die is finished. It is perfectly safe to push “STOP” button at any time.


Target: GTFE64
Output Filename Prefix: [illegible]
Manual
Current Die:
Test Result ✓
# mismatch: 0

	Voltage (V)	Current (A)
AVDD	0.00	0.00E+0
AVDD2	0.00	0.00E+0
DVDD	0.00	0.00E+0
+3V	0.00	0.00E+0
-5V	0.00	0.00E+0

message:

Logic Analysis System HP16500
GPIB card: 0
Pattern Generator: slot D
GPIB address: 7
Logic Analyzer: slot B

Power Supplies HP662X & Keithley 237
GPIB Address for AVDD etc.: 5
GPIB Address for DVDD: 12

Tolerance on the Number of Bad Dice: 10

command file: J:\glastlab\tests\gtfe64\wafer04c\command.txt
template file: J:\glastlab\tests\gtfe64\wafer04c\template.txt
output directory: [illegible]:\glastlab\tests\gtfe64\wafer04c
output file:
log file:

errors in test
Initialize LA: 0
Resetting LA: 0
Loading Command File: 0
Run a Test: 0
Check Output: 0
error message:

error out from AVDD supply
status ✓ code: +0
source:

error out from DVDD supply
status ○ code: 0
source:

PS error
status ○ code: 0
source:

Front panel of waferTest.vi. The left side includes major control buttons and display of test results, and the right does GPIB settings, file names for the test, error outputs, and so on.

Due to slow GPIB connection, it takes about 5 hours to test all 344 die on a wafer in automatic mode. Also, sometimes a computer running LabMaster gets stuck after 3 to 4 hours of continuous test. In that case, you have to stop waferTest.vi and reboot the computer running LabMaster. You should be careful not to do damage on a probe card nor a wafer when reboot the computer. After the reboot, you can resume a test at any die you would like to start with.

3.2 Test Pattern

Both electronicsTest.vi and waferTest.vi reads a command file to build up a test pattern. A command file consists of individual command lines and comment lines. The VI's read it, interpret it line by line, and take an appropriate action corresponding to an each command. Here is an example of a command file. This is a part of the functionality test of HDI for the beam test tracker construction.

```
#
placePattern: RESET 1*10 0
repeat: 256
resetChip:
repeat: 256
#
# test control register
#
addressRC: 31
chipsToRead: 19
checkSum: include
XYcoincidence: use
readLayer: always
registerRC:
#
addressRC: default
registerRC:
*placePattern: CLKS 1*25 0
#
```

The VI's skip and ignore a line starting with a pound ("#"), so that one can you a line for a comment. Otherwise, a command name followed by a colon (":") must be at the beginning of a command line. Some commands require arguments after the colon. Arguments must follow at least one white space following a command name, separated with at least one white space between arguments. The number of arguments may vary depending on a command.

There are three types of commands: a parameter-setting command, a bit-setting command, and an execution command. A parameter-setting command sets a value to a parameter in the VI's, such as a front-end chip address. A bit-setting command generates a piece of an actual bit sequence to append at the end of the entire bit sequence being built, such as a set-control-register command. If an asterisk ("*") is in front of a bit-setting command, a bit sequence generated by the command starts at the same timing as the bit sequence generated by the previous bit-setting command. An execution command tells the VI to initiate an action on the logic analysis system, such as to run a pattern stored in the pattern generator.

All the commands are listed in tables below. Although most commands are self-explanatory, some commands and arguments need an explanation. Explanations for such commands and arguments follows the tables below.

Parameter-setting Command

Command	Argument	Parameter to be set
AddressFE	<address>	Front-end chip address
AddressRC	<address> | default	Controller chip address
CommandOn	right | left | both	VI's internal control flag
ThresDac	<dac-value> {high | low}	GTFE64 control register (*)
CalibDac	<dac-value> {high | low}	GTFE64 control register (*)
Mask	{all | calib | chan | trig} <ch-list>	GTFE64 control register (*)

CtrlDir	left | right | default	GTFE64 control register (*)
ChipsToRead	<nchip>	GTRC control register (*)
CheckSum	include | exclude	GTRC control register (*)
Xycoincidence	use | ignore	GTRC control register (*)
ReadLayer	always | withFastOR	GTRC control register (*)

(*) Note that a command sets a register setting only to an internal variable of the VI. To set the value to an actual target system, registerFE command or registerRC command must be issued after the parameter-setting command.

Bit-setting Command

Command	Argument	Remark
RegisterFE	none	GTFE64/GTRC command
RegisterRC	none	GTFE64/GTRC command
ReadEvt	none	GTFE64/GTRC command
Strobe	none	GTFE64/GTRC command
ResetChip	none	GTFE64/GTRC command
no_op	none	GTFE64 command
ResetFifo	none	GTFE64 command
ClearFifo	none	GTFE64 command
EndRead	none	GTFE64 command
ClearEvt	none	GTRC command
ClockOn	none	GTRC command
SendResetChip	none	GTRC command
SendResetFifo	none	GTRC command
PlacePattern	<line-name> <bit-pattern>	Set a bit pattern on an input line
Repeat	<nstep>	Repeat the last line <nstep> times
Default	<nstep>	Append <nstep> “default” lines
Empty	<nstep>	Append <nstep> “empty” lines

Execution Command

Command	Argument	Remark
runPattern	[readData | readAndDecodeData]	Run a current pattern

3.2.1 “default” for “addressRC” and “ctrlDir” command

If an argument of “addressRC” command or “ctrlDir” command is “default”, then the VI takes a value from a front-panel of the VI and set it to an internal variable of the VI to be set to an actual target system by “registerFE” command or “registerRC” command appearing later. See the previous sections for explanations on how to set it on a front-panel. This is useful for an HDI test, in which GTRC address is hard-wired and control direction should be able to change depending on which side of an HDI the testing system connects to.

3.2.2 “commandOn”

This command is valid only for a GTFE64 test and defines which command line, CMDL or CMDR, of a GTFE64 chip, a command for GTFE64 should appear. Put “left” as an argument for a command to appear on CMDL, “right” for CMDR, “both” for both.

3.2.3 Channels to Mask

To set a trigger mask, a channel mask, or a calibration mask of GTFE64 chips, “mask” command can be used in testing either of a single GTFE64 chip or an HDI. The first argument is a mask type: “trig” to set a trigger mask, “chan” to set a channel mask, “calib” to set a calibration mask, or “all” to set the same mask to all the three masks at once. After the second argument, a list of channels to be enabled should be listed, namely, channels to be set “1” on its mask. A list of channels should be given as an arbitrary combination of a single integer, two integers connected by a hyphen, or three integers connected by hyphens. Two integers connected by a hyphen give a range of channels to be enabled. The first two of three integers connected by hyphens give a range of channels to be enabled, and the third gives the number of channels to be skipped. Here are a few simple examples.

mask: trig 4 19 37	Set 1 to a trigger mask of channels 4, 19, and 37
mask: chan 29-41	Set 1 to a channel mask of channels from 29 through 41
mask: calib 3-19-4	Set 1 to a calibration mask of every 4th channel starting with channel 3 and not exceeding channel 19. In other words, set 1 to a calibration mask of channels 3, 7, 11, 15, and 19.
mask: trig	Set 0 to a trigger mask of all the channels
mask: all 3 5-7 12-23-5	Set 1 to all the masks of channels 3, 5, 6, 7, 12, 17, and 22

3.2.4 “placePattern” command

This command places an arbitrary bit sequence on any specific input line. The first argument is a name of the input line, such as CLKS (sampling clock). Starting from the second argument, a bit sequence can be given as an arbitrary combination of a single integer and two integers connected by “*” (asterisk). An integer following an asterisk gives the number of repetition of the integer in front of the asterisk in a bit sequence. This command is mostly used to control a sampling clock (CLKS) in combination with “*” (asterisk) at the beginning of the command to sample a response for the previous command. A few simple examples are shown here.

***placePattern: CLKS 1*25 0**	Place twenty five “1”s and a single “0” on CLKS
placePattern: RESET 1*10 0	Place ten “1”s and a single “0” on RESET

Note that “0” is at the end of a bit sequence in both examples above. Both of the VI’s may put “padding bits” after the end of the bit sequence specified by this command. A padding bit is an extra bit automatically appended at the end of the bit sequence specified, in order to make the length of the bit sequence equal to the lengths of other lines. The value of the extra bit is taken from the last bit of the bit sequence specified. Therefore, without “0” at the end of a command line, you may have more “1”s than explicitly specified in the command line. To avoid possible extra “1”s, it is always recommended to put a single “0” at the end of this command.

A complete list of input lines is in a following table. All of lines from the pattern generator are connected to a target system being tested, except for two lines: CLKS for a sampling clock and USRSIG for an external use. The line CLKS is connected to a clock input of the state analyzer on the converter board to sample an output pattern from a target being tested. The line USRSIG is not connected to anything, so that you can use it for an external use such as using it as a trigger of an oscilloscope.

Most of those lines are single bit line, on which only “0” or “1” can be placed. Some are a multi-bit line, on which you can specify a positive integer value to it. The range of valid values depends on how many bits are assigned to the line. The number of bits available on a line is also shown in the table below in parenthesis.

Target	Name of input lines	Remark
GTFE64	CLKR, CLKL, CMDR, TACKR, CMDL, TACKL, TRI, TLI, DRI, DLI, ADDR(5), RESET	Connected to a chip
	CLKS	Sampling clock
	USRSIG(8)	For external use
GTRC	RDIN, SE, ADDR(5), RESET, CTLREG, SDI, TOKI, DIN, TRI, CMDI, CLKI, TACKI	
	CLKS	Sampling clock
	USRSIG(8)	For external use
HDI	TOKI, DIN, CMDI, TACKI, CLKI, RESET	
	CLKS	Sampling clock
	USRSIG(8)	For external use

(*) The number of bits assigned to a line is shown in parenthesis, unless only one bit is assigned to the line.

3.2.5 “runPattern” command

When the VI encounters this command, it immediately triggers the pattern generator to run a test pattern. After one run, a test pattern stored in the pattern generator will be discarded. With “readData” as an argument, it arms the state analyzer before the run, takes out data from the state analyzer after the run, and appends it to an output file. With “readDataAndDecodeData” as an argument, it does the same actions as with “readData” option first, and decodes a bit sequence on DOUT line to append it to an internal array of events. The event array is displayed on a front-panel of electronicsTest.vi and written into a log file as an attachment (see below for a log file format). The VI waferTest.vi doesn’t support “readDataAndDecodeData” option yet. With no argument, it just runs a test pattern and does nothing on the state analyzer.

3.2.6 “repeat”, “default”, and “empty” command

These commands are to append a bit sequences on all the lines to insert a waiting time in a test pattern. The length of a waiting time is defined by an integer, the number of clock cycles to wait. Each command has a different set of bit sequences which applies to input lines during the waiting period. The “empty” command inserts “0” on all the lines. The “repeat” command repeats the value at the last clock cycle before this command on each line. The “default” command repeats a “default value” of each line, which is defined as “a value in the previous clock cycle” for ADDR line and “0” on all the others.

3.3 File Formats of Output Pattern and Template Pattern

Both of electronicsTest.vi and waferTest.vi outputs an output pattern into an output file of the same format. An output file is a column-wise ASCII file which contains all the digital outputs from the target system. Columns are delimited by a tab. The number of columns varies depending on what to test (GTFE64, GTRC, or HDI). Here is how it looks in the case of a GTRC test:

```
LABEL   TOKO    DOUT    TOUT    CMDL    CLKL    TACKL
NDATA   8100    8100    8100    8100    8100    8100
0       0       0       0       0       0       0
1       0       1       0       0       0       0
2       0       1       1       0       0       0
…..
```

The first line of an output file contains a column header. In a column header, names of digital outputs are listed, with the first column “LABEL” indicating that this is a column header. In the second line, the number of lines in each column is listed. All columns must have the same number of lines. From the third line, an output pattern starts. The left-most column in this section is a clock count of a sampling clock (“CLKS” in a test pattern). The number starts with 0 and ends with (*n-1)* where *n* is the number listed in the second line of the file (*n=8100* in the example above). On each line, either “0” or “1” is written at each column to show that a digital output of the column was logical high or logical low. Therefore, the example above shows that DOUT became logical high between the second and the third sampling edge of a sampling clock and stayed high at the third sampling clock. It also shows that TOUT was logical high at the second clock and all others stayed logical low. The example must continue on until the clock count in the left-most column reaches 8099. Below is a list of columns listed in an output format.

Target system	Columns listed
GTFE64	TRO, TLO, DROP, DRON, DLOP, DLON, CTLREG
GTRC	TOKO, DOUT, TOUT, CMDL, CLKL, TACKL
HDI	TOKO, DOUT, TOUT

3.4 File Format of Template File

A template file looks similar to an output file. Indeed, an output file of one test can be used as a template file for other test. In addition to an output file format, a template file can include additional features to help test and diagnosis: tolerance in comparison and a comment on a line. Here is an example of a template file for an HDI test.

```
LABEL   TOKO    DOUT    TOUT
NDATA   10150   10150   10150
0       0       0       X       registerRC (addressRC:default, 1st pattern)
1       0       0       X
2       0       0       X
…..
```

Tolerance in comparing an output file with a template file can be introduced in a template file, which includes “X” instead of “0” or “1” in its data contents. Both VI’s compares an output file with a template file bit by bit, if the numbers of lines of both files match. In the comparison, it ignores a bit with “X” in a template file. In other words, no matter what an output file contains at the same location as “X” in a template file, a target system passes a test if there is no other mismatch. So, “X” may be put for bits that you don’t care.

A comment on a line in a template file is useful to diagnose a target system. A comment can be put at the end of any data line. In an example above, the first line has a comment saying “registerRC (addressRC:default, 1st pattern)”. The comment line will be copied into a log file to help diagnosis. Therefore, a comment should be placed at the beginning of each section of a test, which has a separate meaning from other parts of the test. For example, you may place “registerFE (addressFE:13)” at the beginning of a section to test a control register of a GTFE64 chip with address of 13 on an HDI.

3.5 File format of Log File

A log file contains results from a test. An example is shown below. It is not used by the VI’s as an input file. It is only for human eyes to look at for diagnosis purpose.

```
Check results of a data file with a template pattern
  Data file: J:\glastlab\tests\hdi_qa\hdi29\beforeStack_outL.txt
  Template:  J:\glastlab\tests\hdi_qa\testPattern\alltest_tplL.txt
278 mismatch(es) found.
===== list of mismatch(es) =====
LABEL TOKO  DOUT  TOUT
6172:  0(X)   0(1)   0(X)    registerFE (addressFE:24)
6175:  0(X)   0(1)   0(X)
6178:  0(X)   0(1)   0(X)
6181:  0(X)   0(1)   0(X)
…..
 6627:  0(X)   0(1)   0(X)
======== end of log file ========
-------- attachment --------
Number of events were read out during the test above: 2
EVENT ADDR  NHIT  ERROR        TOT   CRC   HitList ([HIT]CHIP:STRIP)
0     2     25    0            205   OK    [0]1:11 [1]2:11 [2]3:11 …..
1     2     1     0            0     OK    [0]20:11
-------- end of attachment --------
```

After file names of an output file of the test and a template file used, the number of mismatches are shown. Following those, there is a section of a list of mismatches. The section lists only lines with a mismatch that the VI found in the comparison with a template file. The left-most column is the clock count of a line with a mismatch an output file, followed by contents of an output file and a comment of the section defined in a template file. In addition, each column contains contents of a template file in parenthesis right next to those of an output file.

At the end of a template file, there is an attachment containing a decoded readout from a GTRC chip. This appears only when either of a GTRC chip or an HDI is tested with a test pattern including "runPattern" command with "readAndDecodeData" option. This attachment is not used for a GTFE64 test. In the attachment, the number of events read out is shown, followed by a list of decode results, which contains event number, a GTRC address, the number of hits, read-out error flag ("1" for error and "0" for no error), a time-over-threshold value, check result of a frame check sequence, and a list of hits. A list of hits contains a hit number in square brackets ("[" an "]"), a chip number, a colon as a separator and a strip number. A chip number ranges from "1" to "25" as "1" means the GTFE64 chip closest to the GTRC chip tested. A strip number runs from "0" through "63" as "0" means the closest strip to the GTRC chip tested.

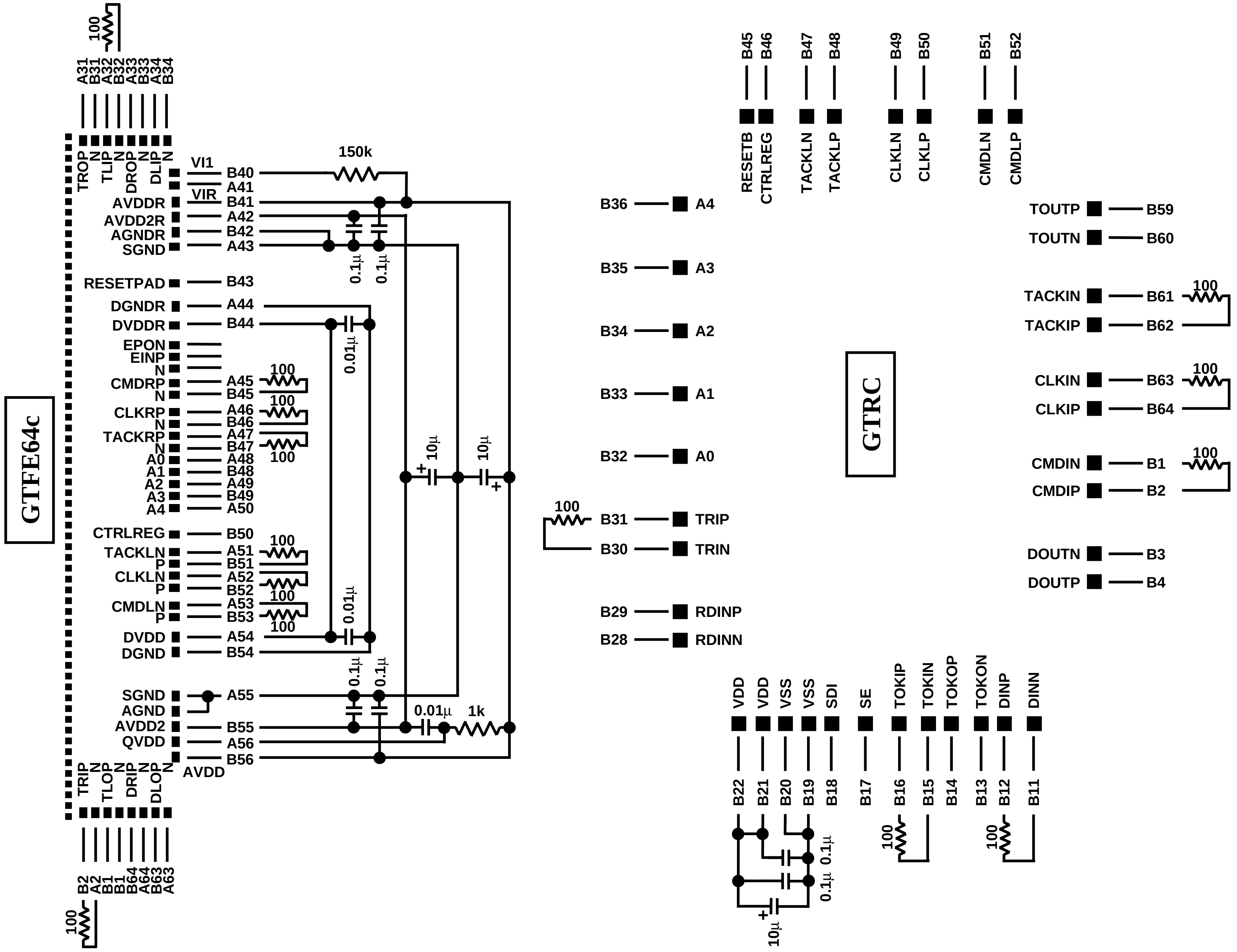
Circuit Diagrams of Electrical Test System for the GLAST Readout Electronics

Masaharu Hirayama

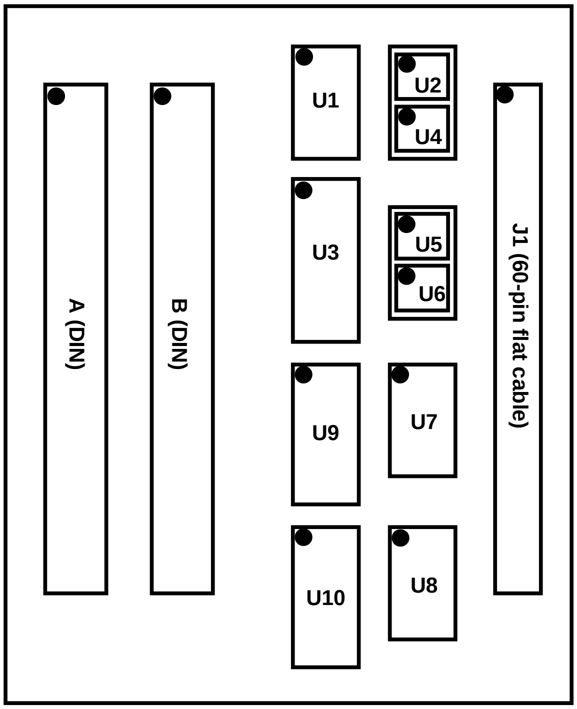
Santa Cruz Institute for Particle Physics

January, 2000

Passive components on probe cards



Interface board for GTFE64c test (1/3)

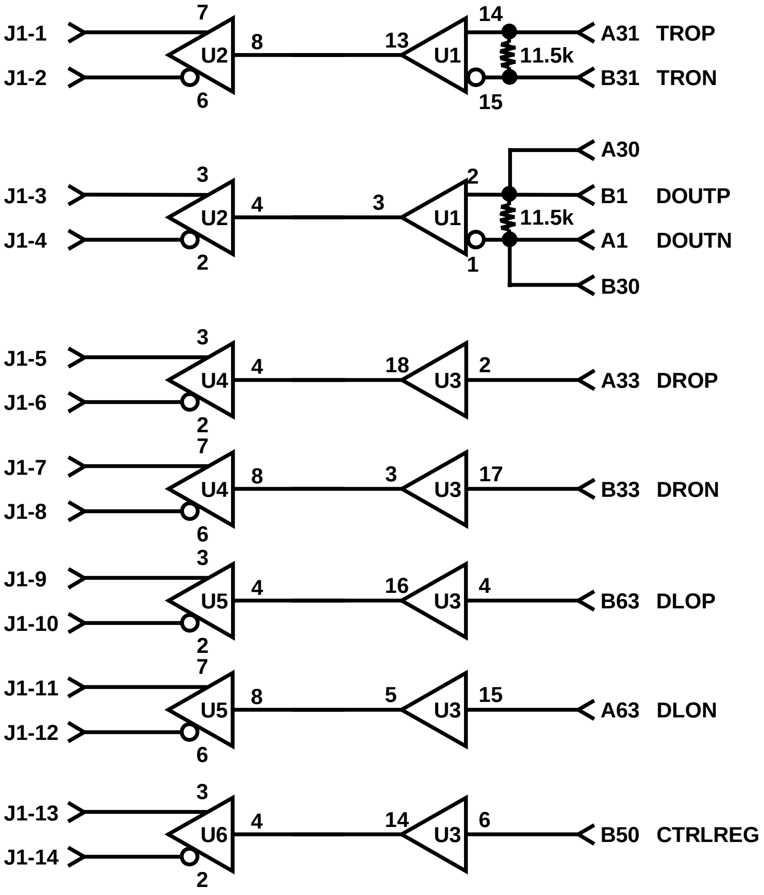


U1 : DS90C032		+5V : pin 16	GND : pin 8
U2 : DS90C401		+5V : pin 1	GND : pin 5
U3 : 74LS244		+5V : pin 20	GND : pin 10
U4-U6 : DS90C401		+5V : pin 1	GND : pin 5
U7-U8 : DS90C032		+5V : pin 16	GND : pin 8
U9 : 74LCX240		+3V : pin 20	GND : pin 10
U10 : 74LCX244		+3V : pin 20	GND : pin 10

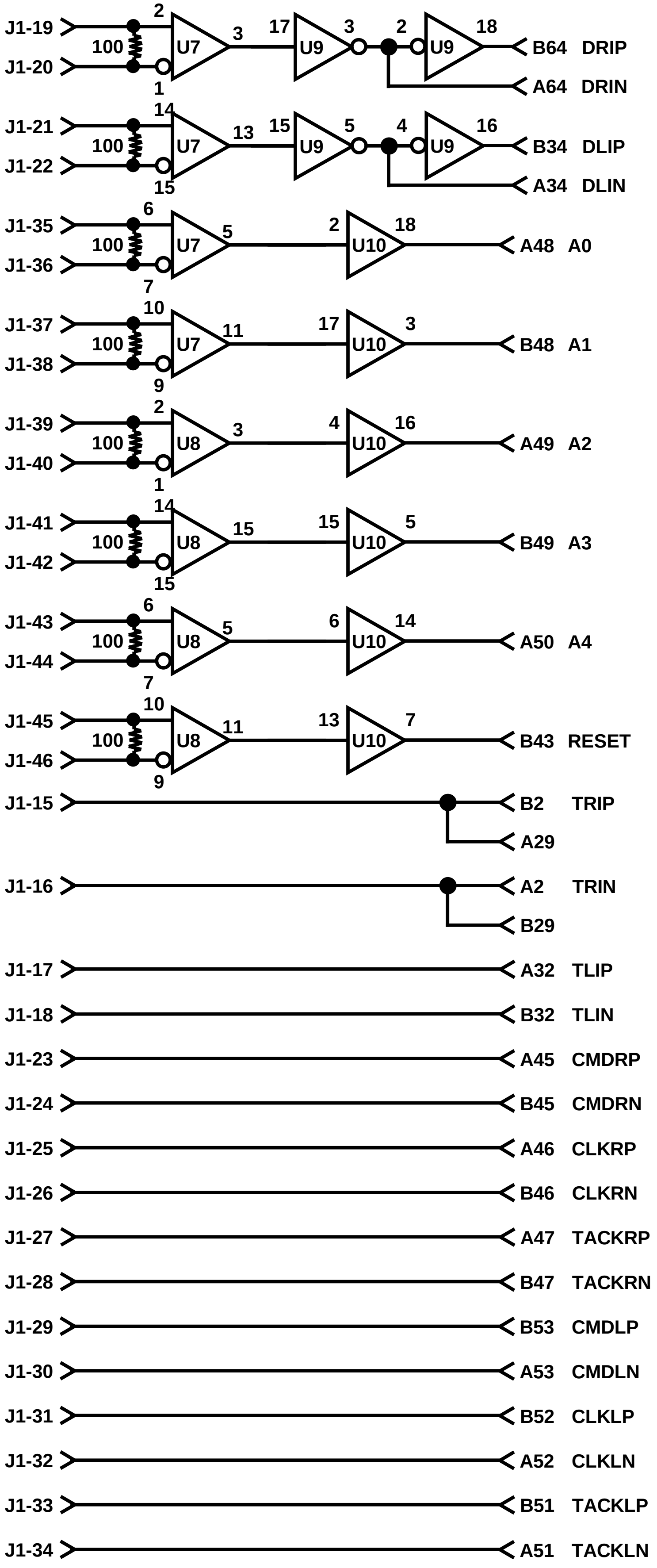
U1-U10 : 0.1μF mounted on power pin

U1, U7, U8 : pin 4, pin 12 connected to GND

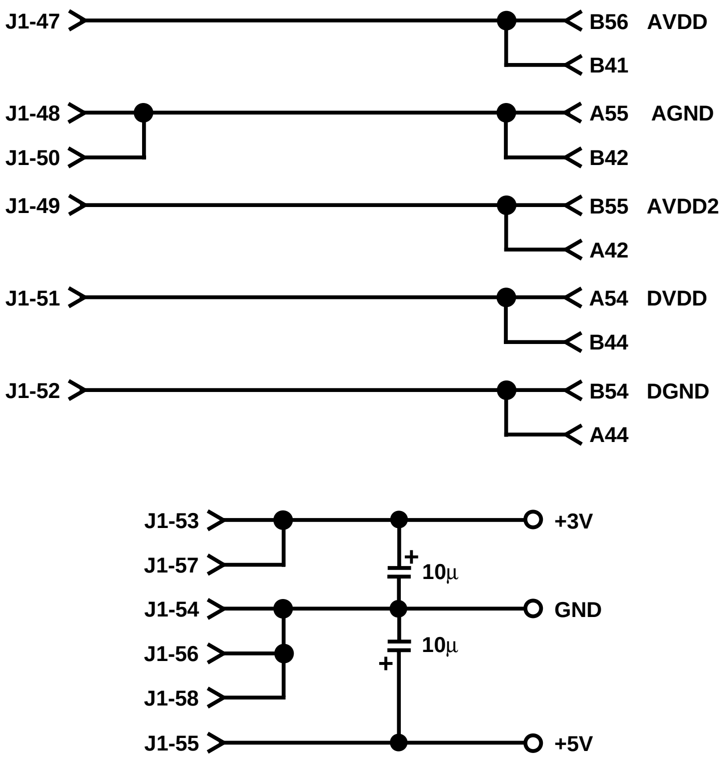
U3, U9, U10 : pin 1, pin 19 connected to GND



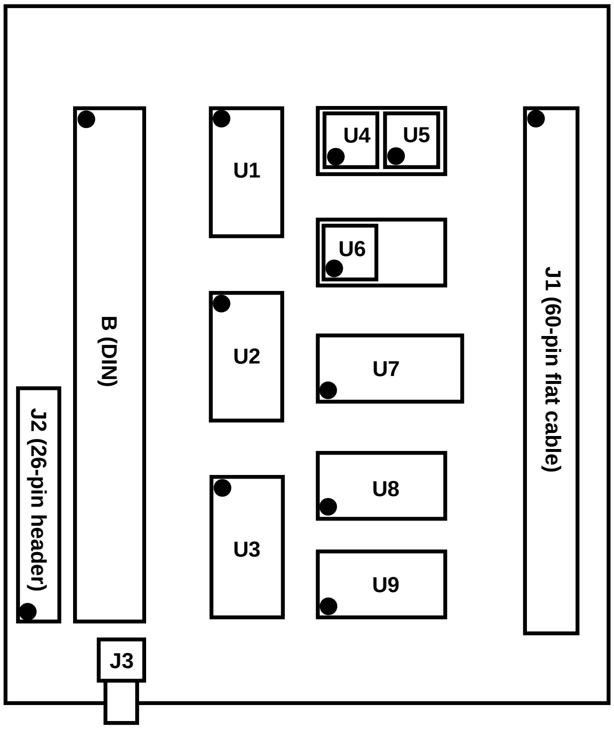
Interface board for GTFE64c test (2/3)



Interface board for GTFE64c test (3/3)



Interface board for GTRC test (1/3)

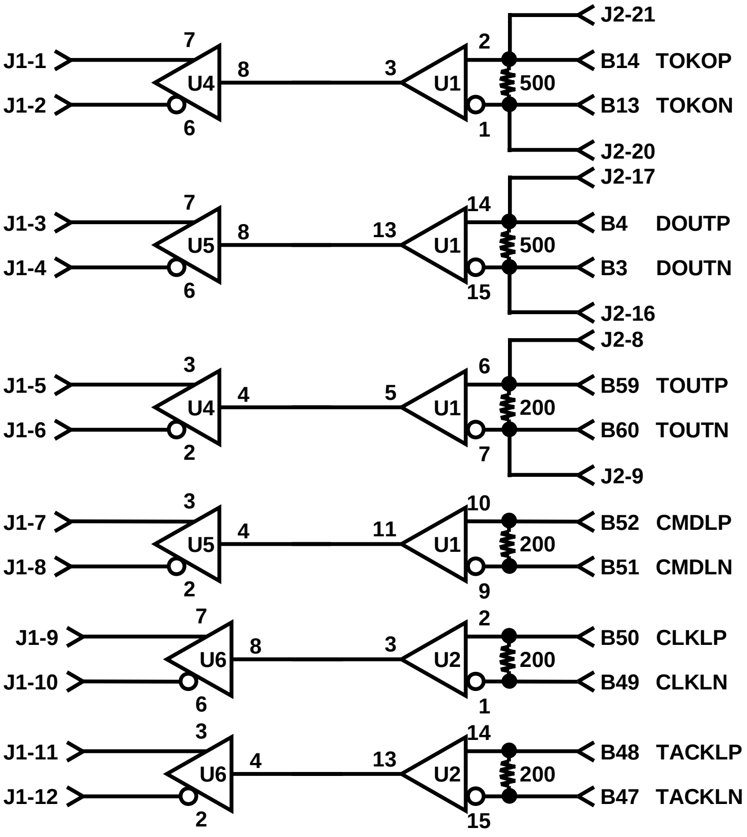


U1-U2 : DS90C032	+5V : pin 16	GND : pin 8
U3 : 74LCX244	+3V : pin 20	GND : pin 10
U4-U6 : DS90C401	+5V : pin 1	GND : pin 5
U7 : 74LCX240	+3V : pin 20	GND : pin 10
U8-U9 : DS90C032	+5V : pin 16	GND : pin 8

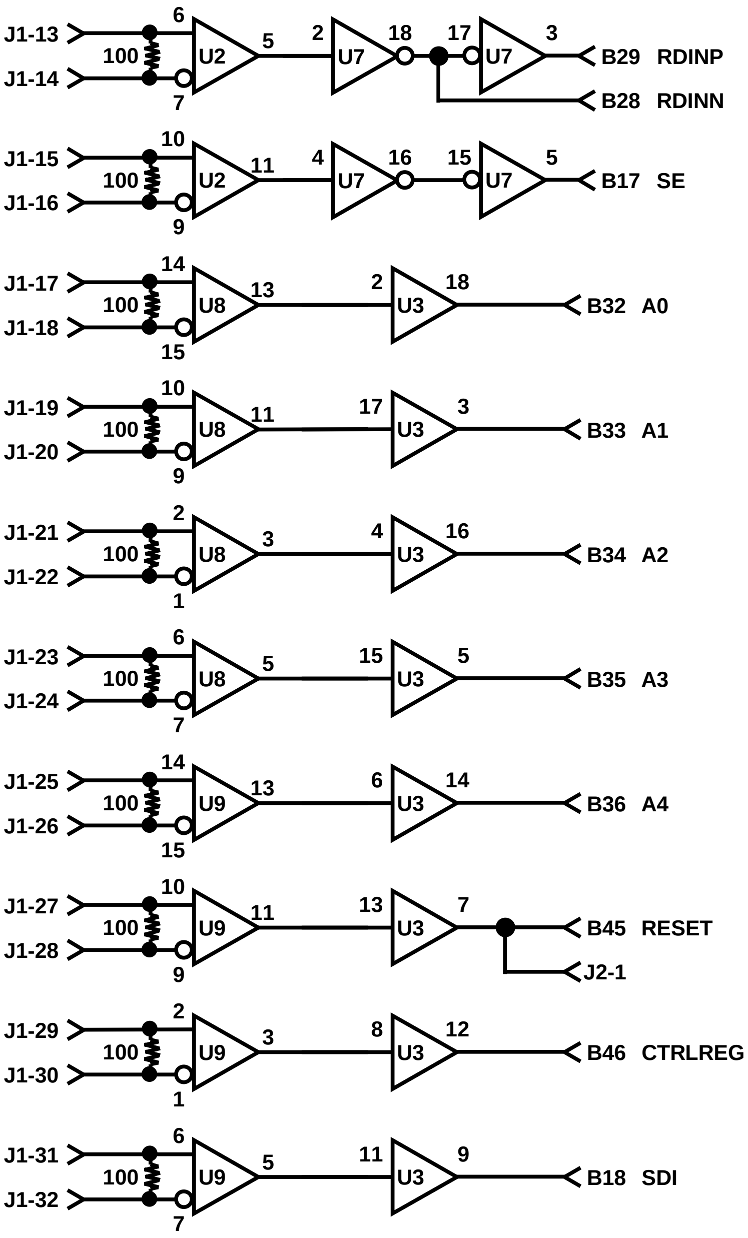
U1-U9 : 0.1μF mounted on power pin

U1, U2, U8, U9 : pin 4, pin 12 connected to GND

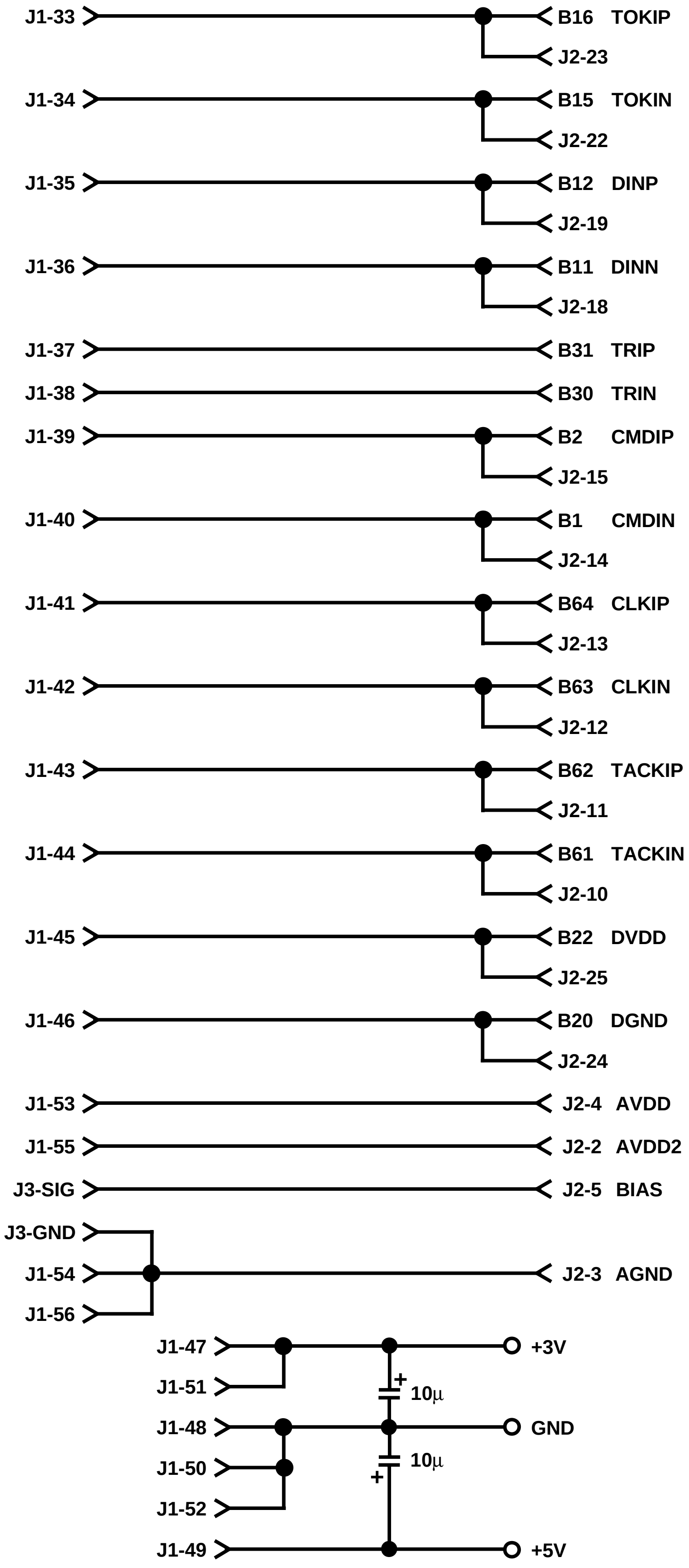
U3, U7 : pin 1, pin 19 connected to GND



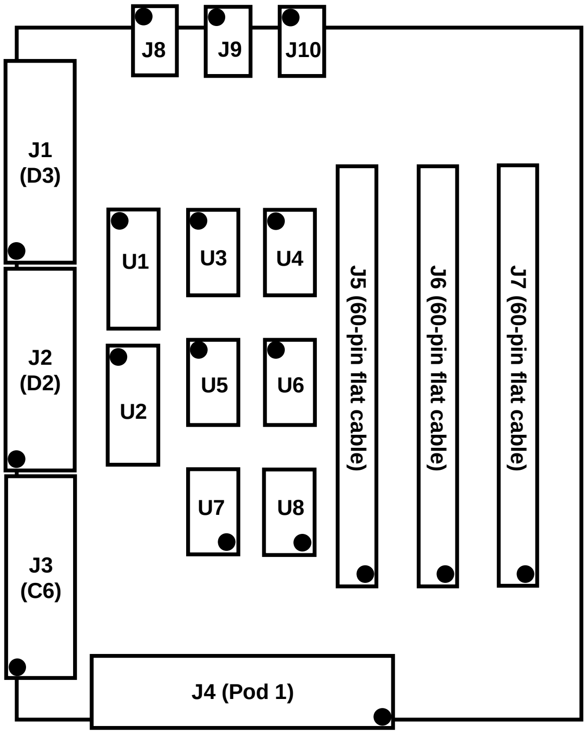
Interface board for GTRC test (2/3)



Interface board for GTRC test (3/3)



Converter board (1/3)



U1-U2 : 74LCX244 +3V : pin 20 GND : pin 10

U3-U6 : DS90LV047 +3V : pin 4 GND : pin 5

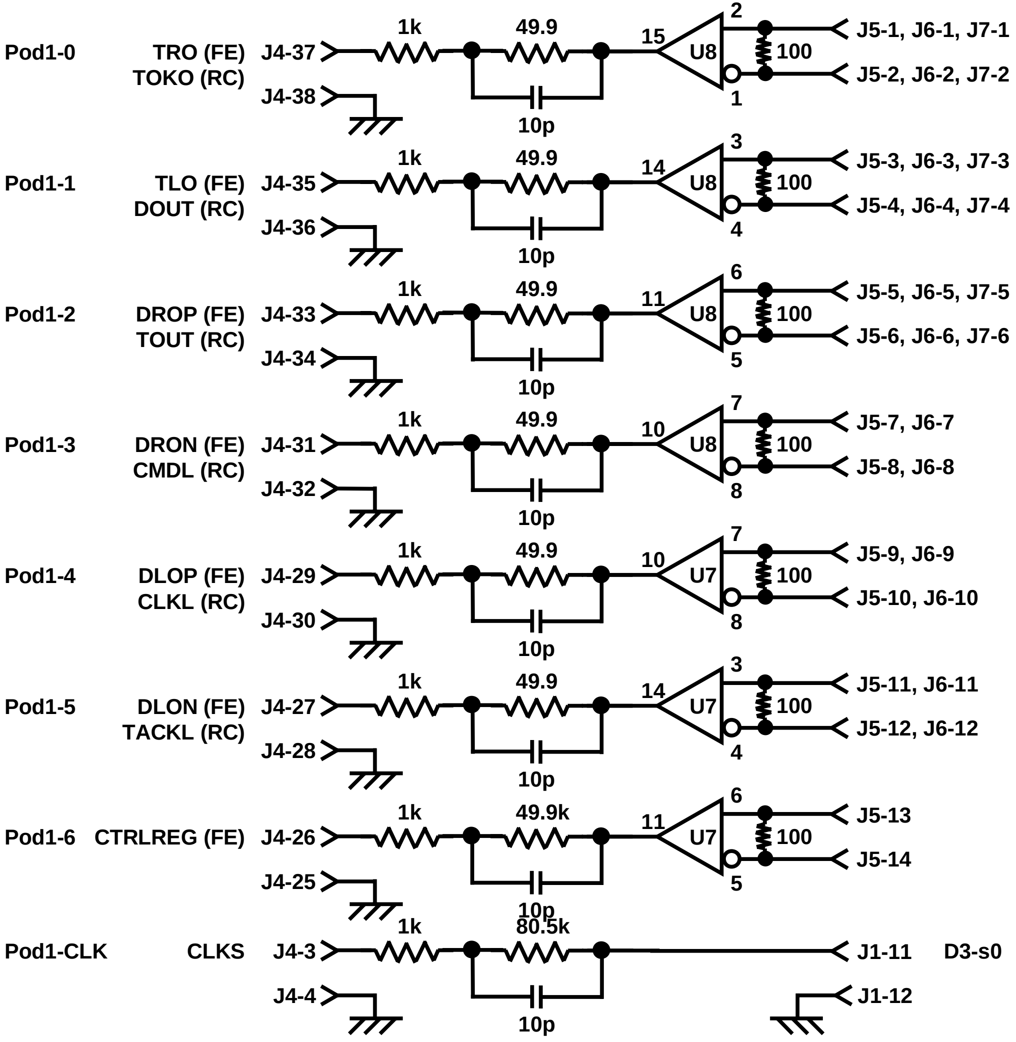
U7-U8 : DS90LV048 +3V : pin 13 GND : pin 12

U1-U8 : 0.1μF mounted on power pin

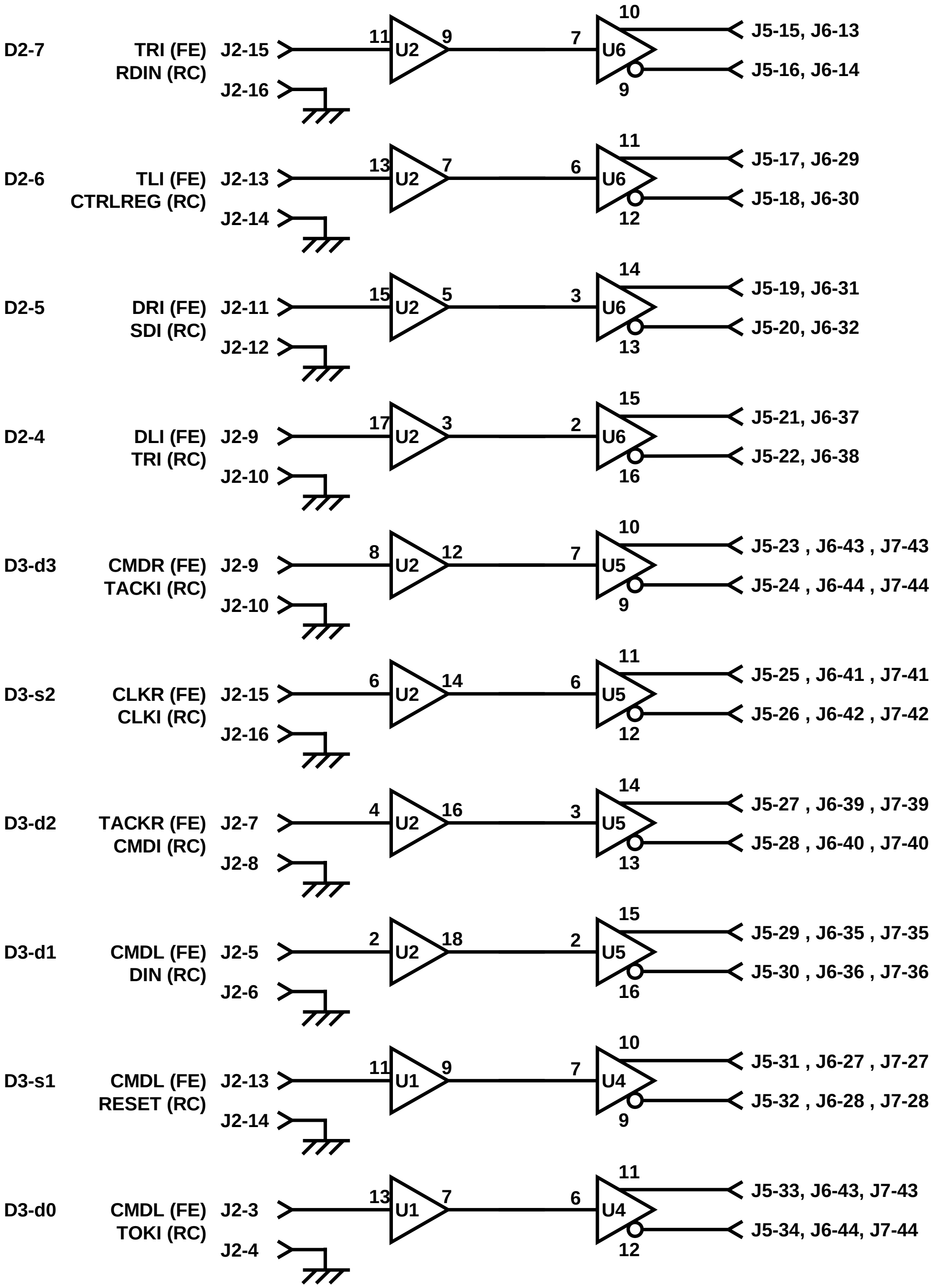
U1-U2 : pin 1, pin 19 connected to GND

U7-U8 : pin 16 connected to +3V, pin 9 to GND

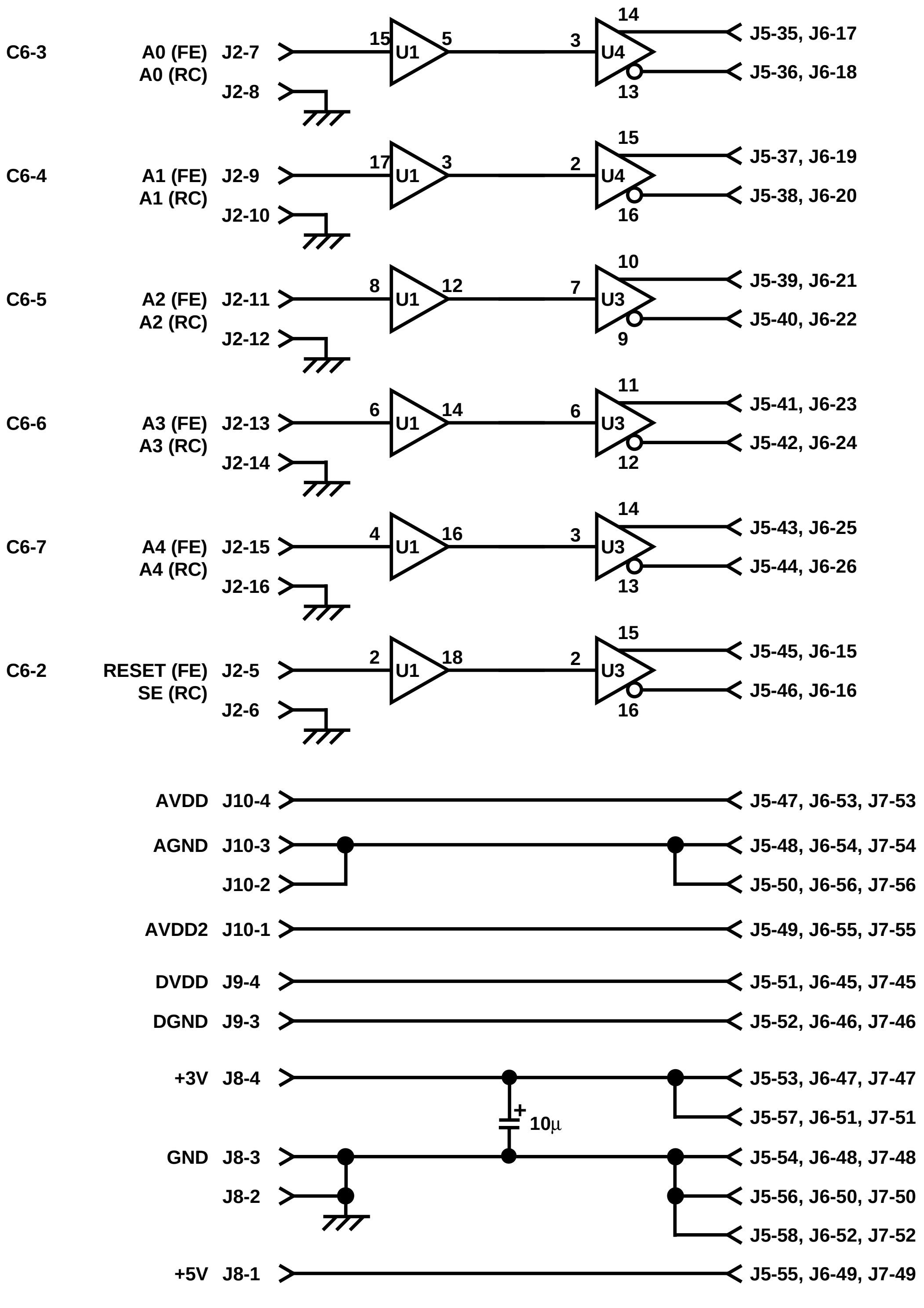
U3-U6 : pin 1 connected to +3V, pin 8 to GND



Converter board (2/3)



Converter board (3/3)



C6-3
A0 (FE)
A0 (RC)
J2-7
J2-8
15
U1
5
3
U4
14
13
J5-35, J6-17
J5-36, J6-18
C6-4
A1 (FE)
A1 (RC)
J2-9
J2-10
17
U1
3
2
U4
15
16
J5-37, J6-19
J5-38, J6-20
C6-5
A2 (FE)
A2 (RC)
J2-11
J2-12
8
U1
12
7
U3
10
9
J5-39, J6-21
J5-40, J6-22
C6-6
A3 (FE)
A3 (RC)
J2-13
J2-14
6
U1
14
6
U3
11
12
J5-41, J6-23
J5-42, J6-24
C6-7
A4 (FE)
A4 (RC)
J2-15
J2-16
4
U1
16
3
U3
14
13
J5-43, J6-25
J5-44, J6-26
C6-2
RESET (FE)
SE (RC)
J2-5
J2-6
2
U1
18
2
U3
15
16
J5-45, J6-15
J5-46, J6-16
AVDD
J10-4
J5-47, J6-53, J7-53
AGND
J10-3
J5-48, J6-54, J7-54
J10-2
J5-50, J6-56, J7-56
AVDD2
J10-1
J5-49, J6-55, J7-55
DVDD
J9-4
J5-51, J6-45, J7-45
DGND
J9-3
J5-52, J6-46, J7-46
+3V
J8-4
J5-53, J6-47, J7-47
J5-57, J6-51, J7-51
+
10µ
GND
J8-3
J5-54, J6-48, J7-48
J8-2
J5-56, J6-50, J7-50
J5-58, J6-52, J7-52
+5V
J8-1
J5-55, J6-49, J7-49