AutoAdjust Routines

Objectives

AutoAdjust is a function that enables the system to meet its accuracy specifications for all time- related measurements. This chapter describes what the AutoAdjust routines do within the system.

NOTE **DO NOT troubleshoot from AutoAdjust failure messages. Instead, troubleshoot from Diagnostic failure messages only.**

Amplitude vs. Timing Routines

Amplitude AutoAdjust

In **amplitude** AutoAdjust routines, voltage levels in the analog and digital subsystems are compared with reference levels that were stored during calibration, and correction constants are used to compensate for gain and offset errors in the measurement circuits.

Timing AutoAdjust

In **timing** AutoAdjust routines, a time interval counter and delay lines in the digital subsystem are used to de-skew[1] timing signals within a module and between modules. This process is responsible for the driver and receiver edge placement accuracy specifications.

How AutoAdjust Constants are Stored

After running AutoAdjust, correction constants are stored on the system controller’s disk and downloaded to memory on the Module Control Cards.

1. The term **de-skew** means to synchronize two signals such that their edges occur at the same time.

System Reference Measurements

During calibration, reference measurements are made as follows:

1 The reference voltages (VRefs) and a reference resistor on the ASRU Card are measured by an external multimeter.

2 The meter reading values are sent to the controller.

3 The controller verifies that the values are within defined limits. This ensures that a false reading has not occurred due to a malfunction in the AutoAdjust reference circuit or the multimeter.

4 If the reference values are within the limits, they are stored in EEPROM on the ASRU Card and on the system disk drive.[1]

These reference values are the basis of all amplitude- related AutoAdjust routines. Because the reference values are so important, they are updated periodically (on a preventive maintenance schedule) to compensate for aging of circuit components.

1. To read the stored reference values, look in the ASRU_References file in the Keysight subfolder diagnostics/th1/

Amplitude AutoAdjust

There are many amplitude-related AutoAdjust routines; following is one example. Test 4250, Detector (Integrator) AutoAdjust, compensates for gain and offset errors in the detector. Refer to Figure 12-1 for the following explanation:

1 Setup: K2301 open, K2302 open, K2303 closed, K2304 closed. This grounds both inputs of the range amplifier.

2 The detector takes a measurement on all ranges of the range amplifier. These ground readings are stored in the AutoAdjust table (DRAM on the Module Control Card.)

3 Setup: K2301 open, K2302 closed, K2303 closed, K2304 open.

4 Setup: +VRef to low range, range amplifier to low range.

5 The detector measures +VRef and stores the +VRef reading (range 1) in the AutoAdjust table.

6 Setup: -VRef to low range, range amplifier to low range.

7 The detector measures -VRef and stores the -VRef reading (range 1) in the AutoAdjust table.

8 Steps 4 through 7 are repeated for the remaining ranges (9 ranges total).

9 The gain for all ranges is calculated and stored in the AutoAdjust table. The gain is expressed as volts per ADC count. The gain in calculated as follows:

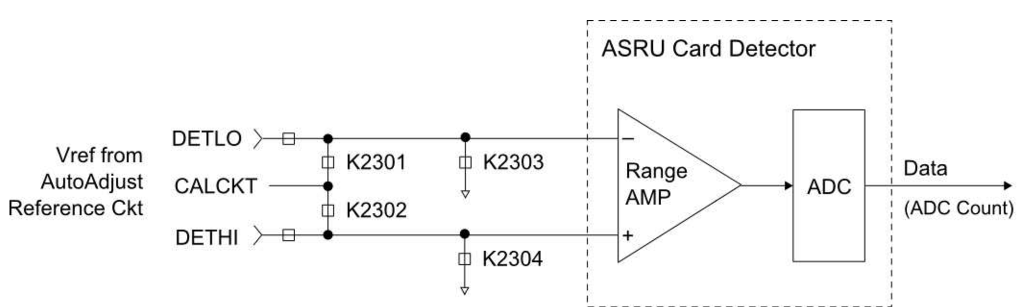
$$\text{Gain} = \frac{+\text{VRef} - (-\text{VRef})}{+\text{VRef reading} - (-\text{VRef reading})}$$

Where: +VRef and –VRef are the actual values of the VRefs that were measured and stored in EEPROM during calibration.

These gain values are used in all subsequent measurements using th detector (in the integrator mode). The actual value of voltage measured by the detector, then, is calculated as follows (the ground readings were taken in step 2):

Active Value = (ADC Count – Ground Reading) * Gain

Figure 12-1 Amplitude AutoAdjust



Timing AutoAdjust

Timing AutoAdjust routines ensure that drivers and receivers meet their edge placement accuracy specifications. The timing AutoAdjust routines first de-skew all drivers and receivers within each module. Then, after each module is de-skewed, the testhead as a whole is de-skewed; that is, the drivers and receivers are de-skewed across module boundaries.

These procedures are explained in the following topics:

- Module De-Skew
- Testhead De-Skew

In all timing AutoAdjust routines the time interval counter (TIC) on the Module Control Card is used to make the timing measurements.

Module De-Skew

Each driver and receiver has its own, independent delay line. The delay lines are located in the format chip. Each delay line has seven serial segments that can be individually or collectively selected. The shortest segment is approximately four nanoseconds; the longest is approximately 164 nanoseconds.

The module timing AutoAdjust routine includes several steps. First, the delay time of a channel is measured with no delay segments selected. This yields the propagation time of the channel. Then, the delay time of the channel is measured with each delay segment selected one-at-a-time. By subtracting the propagation time from each delay segment measurement, the delay time of each segment is calculated. Likewise, these measurements are repeated for every driver channel and every receiver channel on every Pin Card in the module. All of these propagation and delay time measurements are stored in the AutoAdjust table.

Following is a description of how the stored propagation and delay times are used to de-skew a module.

All timing in a module is referenced to the tester clock (TCLK). Suppose, for example, that all drivers are programmed to drive at the same time. The system, knowing the different propagation times of each channel, selects delay segments in each channel such that all channels are delayed by an amount of time that is greater than the longest propagation time of any individual channel. Thus, all drivers drive at the same time, called T0.

Now suppose that a driver is programmed to drive after (delayed from) the previous drivers. This delay time is added to T0 and extra delay segments are selected for this drive channel. Of course, this technique also applies to receiver channels.

Testhead De-Skew

Testhead de-skew is used in multi-module testheads. One module is designated the main module. (This designations are made by the system; the programmer need not be concerned with this.) The main module sends its TCLK signal to all other modules via the high-speed link. The CPUs in all other modules are synchronized to the TCLK signal from the main module.

A complication of using the high-speed link to synchronize the modules is that the link cable introduces propagation delay to the TCLK signal as it travels from module to module around the loop. The value of the cable delay is a known constant, but the link interface circuitry also contributes circuit delay. An AutoAdjust routine measures the circuit delay between modules by sending a calibration pulse around the loop. The circuit delay is added to the cable delay to determine the total link propagation delay, and this value is stored in the AutoAdjust table. Then a programmable delay line in each module's TCLK generator (in the phase-lock loop) is programmed to compensate for the link propagation delay. Thus, the TCLK signal is de-skewed across module boundaries.

CAUTION

If a good file from a passing AutoAdjust test is not available, constants are downloaded from a default file. Do not delete these default files from the system or the test accuracy will be severely impaired.

How AutoAdjust Constants are Stored

When running AutoAdjust, the resulting correction constants may or may not be saved to the controller's disk, and may or may not be downloaded to the Module Control Cards' memory. These actions depend on how AutoAdjust was run and whether there were any errors.

From the AutoAdjust menu in Diagnostics, you can run AutoAdjust in three ways:

1. **AutoAdjust All** — Press function key F6 to run AutoAdjust on all cards in all modules.

   NOTE **This is the only way to include inter-module timing tests in the AutoAdjust test suite.**

2. AutoAdjust one or more modules — Enter one or more module numbers in the **All Tests** field.
3. AutoAdjust one or more cards — Enter one or more module numbers and slot numbers in the **Control Card**, **ASRU Card** and **Pin Cards** fields.

Steps 1, 2 and 3 all happen during the AutoAdjust process. The reason for step 3 is to ensure that good (valid) constants are always in memory on the Module Control Cards. If correction constants are saved to the disk, they are saved in directory `$AgilentICT_ROOT/ diagnostics/th1/cal_B` or `$AGILENT3070_ROOT/ diagnostics/th1/cal_B`. Under this directory are separate correction constant files for every card in every module in the testhead.

If an AutoAdjust test fails (errors are generated), and the constants are saved to the disk, the data is not written over the original file, but saved in a new `bad<xxx>` file (`<xxx>` = card designation).

Table 12-1 describes what AutoAdjust does with each of these methods depending on whether AutoAdjust errors resulted from the tests.

Table 12-1 How AutoAdjust constants are stored

AutoAdjust All	AutoAdjust one or more Modules	AutoAdjust one or more Cards
No Errors: 1 Writes new constants to memory on all Module Control Cards during AutoAdjust. 2 Saves new constants over the original files on disk. 3 Downloads new constants from disk to memory on all Module Control Cards.	No Errors: 1 Writes new constants to memory on only the Module Control Cards in the specified modules during AutoAdjust. 2 Saves new constants over the original files on disk for the specified modules only. 3 Downloads new constants from disk to memory on the Module Control Cards in the specified modules only.	No Errors: 1 Writes new constants to memory on only the Module Control Cards in the specified modules during AutoAdjust. 2 Does not save to the disk. 3 Does not download to the Module Control Cards.
Errors: 1 Writes new constants to memory on all Module Control Cards during AutoAdjust. 2 Saves new constants to `bad<xxx>` files on disk. 3 Downloads the last good constants from disk to memory on all Module Control Cards.	Errors: 1 Writes new constants to memory on only the Module Control Cards in the specified modules during AutoAdjust. 2 Saves new constants to `bad<xxx>` files on disk. 3 Downloads the last good constants from disk to memory on the Module Control Cards in the specified modules only.	Errors: Same as No Errors above.