KEITHLEY
A Tektronix Company

Application Note Series

Number 2614

Creating Scaleable, Multipin, Multi-Function IC Test Systems Using the Model 2602B System SourceMeter® Instrument

Introduction

Consumer electronics like tablets, smart phones, digital cameras, and portable entertainment systems are becoming ever smaller, faster, and less expensive, and the time to market for such new products is shorter than ever. The semiconductor, and passive and active component industries are continuously pushing their development processes to higher levels of integration and complexity to keep pace. This push to put more circuitry in less space has resulted in the integration of analog, digital, and even RF circuitry into complete Systems-on-a-Chip (SoC). Discrete component manufacturers are similarly consolidating multiple components into a single chip for higher density.

In general, as the complexity of the device under test (DUT) grows, so does the amount and sophistication of testing it. For electrical and electronic component manufacturers, who are under constant pressure to lower their Cost of Test (COT) as part of reducing their overall manufacturing costs, this can be a major issue. While increasing test throughput is critical to reducing the COT, it's only part of the story. Reducing the COT generally means testing more devices in less time using less factory floor space. It also means using and re-using test equipment more efficiently.

What are the implications of these requirements of reducing the COT for test system design?

- Higher pin counts and greater test complexity demand packing more sourcing and measurement channels into the space allocated for the test system, making system density increasingly critical.
- The pace of new component development makes it impractical, if not impossible, to design a special system for every new component. Flexible, scalable, and reconfigurable systems are needed to test multiple device types and to handle new devices with minimal system redesign.
- Using commercially available test equipment wherever possible, rather than custom equipment, maximizes equipment reusability.
- Multichannel test systems with parallel test capability offer significant throughput advantages by allowing different sections of a complex IC or SOC to be tested simultaneously.
- Throughput can also be enhanced by using instruments that allow downloading complete test sequences into instrument memory, freeing the system controller for other functions.

Until recently, there were few solutions optimized for multichannel source-measure applications—test engineers were forced to choose from:

- Relatively simple, but slow instrument-based systems using PC control.
- Fast, but complex instrument-based systems using custom trigger, digital I/O, and communication controllers.
- Highly integrated and fast mainframe-based systems, which are often neither cost-effective nor space-effective solutions.

Keithley's Series 2600B System SourceMeter® SMU instruments offer a fourth alternative, which combines the scalability and flexibility of rack-and-stack instruments with the integration and high throughput of mainframe-based systems. They expand upon the capabilities of their predecessor, Keithley's industry leading Series 2400 SourceMeter® SMU instruments. The Series 2400 SourceMeter instruments are designed for reliable operation in high speed, non-stop production environments, and they continue to be the ideal solution for many test applications, but they have their limitations when it comes to multiple source-measure channel systems. This Application Note will show how the Series 2600B addresses many of the requirements of complex multipin, multifunction IC testing using a digital-to-analog converter (DAC) test as a practical example.

Embedded Test Script Processor and TSP-Link

Figure 1 shows how a multichannel production test system can be implemented using Model 2602B System SourceMeter instruments.

Together, Keithley's new high speed SMU (source-measure unit) design, embedded Test Script Processor (TSP™), and TSP-Link bus provide the Series 2600B with the speed necessary to meet the most demanding throughput requirements. The Test Script Processor is a scalable embedded computer. TSP-Link is an inter-unit communication and trigger synchronization bus—essentially an external backplane. With TSP-Link, systems integrators can connect multiple Series 2600B SourceMeter® instruments and program them as if they are housed in the same chassis. A simple programming interface allows creating powerful, high speed, multichannel tests quickly. Since there is no mainframe to restrict them, users can create seamlessly integrated test systems with up to 32 nodes. In this example, each node is a Model 2602B System SourceMeter instrument, so the system can be scaled up to 64 SMU channels. By packing two channels into a 2U, half-rack chassis, the Series 2600B offers the highest density available in any SMU-based system to address

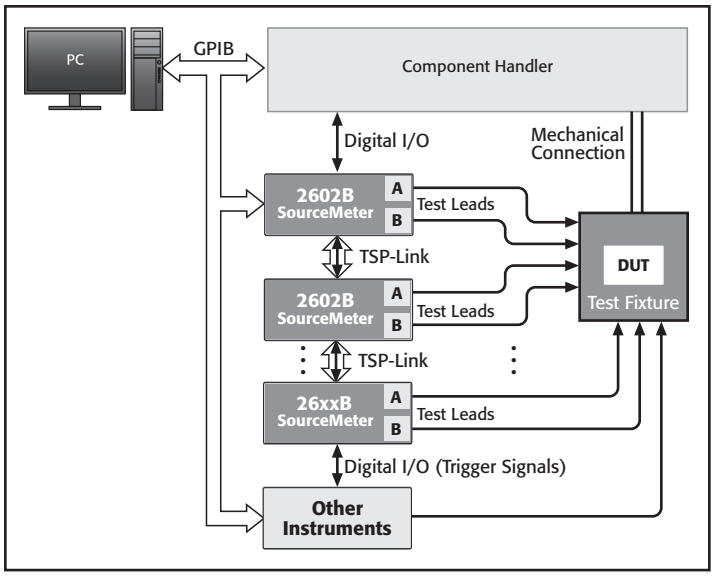

Figure 1. Multichannel test system implemented using Series 2600B System SourceMeter instruments.

growing pin counts and keep rack space manageable. These features will support the development of optimized systems at significantly lower cost per channel and overall smaller capital investments.

Communications between test equipment and the system controller can be a significant throughput bottleneck. It's common practice to perform command and data transfers while the prober or handler is performing mechanical operations to avoid consuming valuable test time. However, as test systems grow increasingly complex, requiring more controller/instrument interaction, this can be difficult to accomplish.

Imagine being able to download an entire test program into one of the instruments in the system, which would then control its own operation and that of the other instruments via a high speed interface. This is exactly what the Test Script Processor and TSP-Link make possible with Series 2600B SourceMeter instruments, as shown in ***Figure 1***. TSP scripts (programs) can be downloaded into either volatile or nonvolatile memory of one "master" unit, which then controls its own operation and that of all of the "slave" units connected to it as though they were a single instrument. This capability frees the system controller to interface with other instruments in the rack more frequently, thereby increasing the overall system throughput. Furthermore, the Series 2600B has very deep memory, which can also reduce how often the host controller needs to interact with the 2600B-based system. Although memory allocation varies with how the SourceMeter instrument is being used, script memory is guaranteed to hold a minimum of 50,000 lines of code and as much as 250,000 lines of code. Each SMU has two nonvolatile data buffers, which can hold from 100,000 to 150,000 readings, depending on the selected data options. Additional volatile memory can provide significantly more data storage, depending on the specific application.

In ***Figure 1***, the master and slave units are connected via TSP-Link, which combines a high speed, serial communication bus and hardware trigger lines into a unique inter-instrument real-time control and data exchange interface. TSP-Link is a peer-to-peer interface, designed to provide communications only between instruments, not between instruments and a host computer. Although some of the familiar host interfaces, such as USB and FireWire, have higher raw bandwidths than the TSP-Link bus, they require sophisticated data transfer protocols and other compatibility features to interact with the host computer. Therefore, they tend to impose high overhead on short messages, which are common in instrument control. As a result, these host interfaces end up being similar in speed to GPIB for sending small amounts of information back and forth. TSP-Link is limited to inter-instrument communications, so it can operate with a simpler message exchange protocol and much lower overhead, resulting in speeds approximately ten times faster than conventional GPIB.

Testing a Generic IC Component

Figure 2 shows a generic IC component. Depending on the component, it can have any combination of analog inputs and outputs, including one or more power inputs. ***Figure 2*** identifies some of the applied signals and measurements that may be required at the inputs and outputs. Some of the measurements require instruments such as counter/timers, gain/phase meters, impedance analyzers, LCZ meters, spectrum analyzers, or RF power meters, but there are many more source and measure requirements that are easily or best satisfied with a SourceMeter instrument.

Source-measure units are an all-in-one solution for current-voltage (I-V) characterization with the combined functionality of a precision power supply, high precision DMM, and electronic load. Keithley pioneered the development of individual, compact bench-top SMU instruments and is the leading supplier of these instruments today.

A SourceMeter instrument also has significant sweep and pulse capabilities. Therefore, given that many measurements perceived as AC measurements can be performed using pulsed or swept DC, it provides a universal analog I/O pin for a wide range of test needs. For our generic IC example, this means that, while additional equipment may be required to test some of today's mixed signal devices fully, one or more SourceMeter instruments are often all that's needed.

Digital-to-Analog Converter Test Example

To demonstrate the power and flexibility of the Series 2600B in a multichannel, multifunction test application, let's look at a possible test sequence for an 8-bit multiplying digital-to-analog converter (DAC). ***Figure 3*** illustrates the test setup for this application. Each instrument connected by TSP-Link is assigned a unique Node Number, similar to a GPIB address. The DAC example uses two nodes (1 and 2), which each have two SMUs (A and B). Node 1 is the "master" node and Node 2 is the "slave"

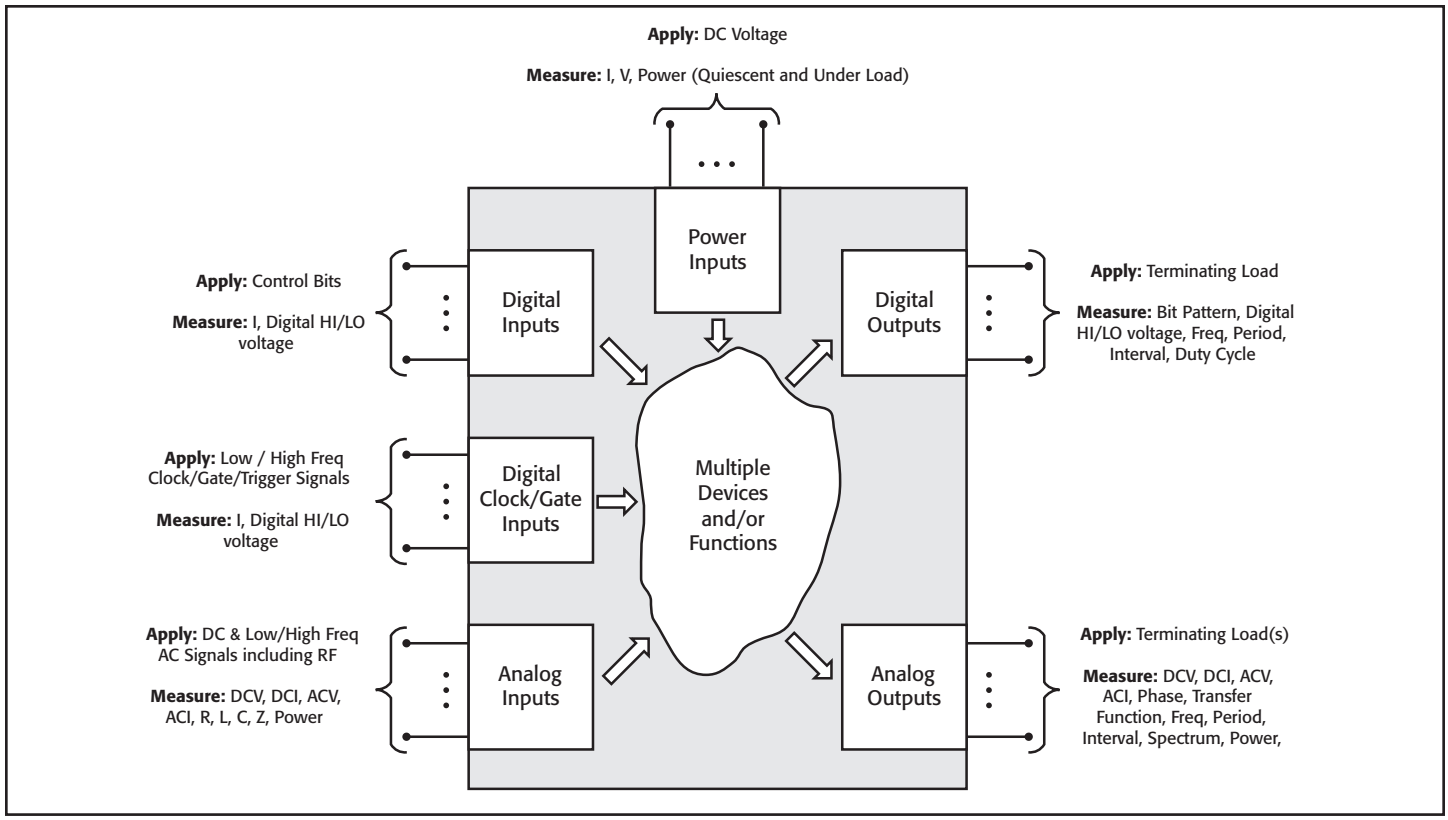

Figure 2. Complex ICs can have multiple analog and digital inputs and outputs. Testing may require multiple measurements with multiple stimuli.

node. The controller sends commands or downloads complete test scripts to the master unit via GPIB (or RS-232), and the master sends commands to the slave via TSP-Link.

The DAC requires DC power and a reference voltage. Node 2 SMU A will provide +15VDC to power the DAC. It will measure the actual supply voltage and the current drawn by the DAC when all digital inputs are set low and again when all digital inputs are set high. Node 2 SMU B will provide +10VDC for V_{REF}. It will measure the input resistance at this terminal when all digital inputs are set high.

The DAC has two complementary current outputs. While one might normally connect a terminating resistance to the output to convert the current to a measurable voltage, the SourceMeter instrument makes it possible to measure the current directly. The levels measured at these outputs are proportional to V_{REF} scaled by a factor that depends on the value of the digital input. I_{OUT1} is maximum and I_{OUT2} is minimum when the digital inputs are all high. Conversely, I_{OUT1} is minimum and I_{OUT2} is maximum when the digital inputs are all low. The outputs will be measured as a function of the digital inputs and their linearity characteristics will be evaluated.

Linearity measurements on DACs are typically performed in a static state. An input code is applied to the digital inputs of the converter, and if necessary, the converter is strobed to pass the data through the converter. The DAC in our example doesn't require a strobe. The Node 1 digital I/O (bits 1 through 8) will provide the digital inputs to the DAC. Six remaining bits of Node 1 DIO and all 14 bits of Node 2 DIO are available for triggering or for digital control. As shown in ***Figure 3***, the Node 1 digital port could interact with the component handler to control binning operations. After the input code is applied, the output of the DAC is allowed to settle, then it's measured several times to eliminate the presence of noise, and an average output current or voltage is recorded. After an appropriate settling time, Node 1 SMU A will measure I_{OUT1}, and Node 1 SMU B will measure I_{OUT2}. The SourceMeter instrument's built-in digital filters will automatically average five measurements for each output reading.

There are several linearity parameters to be evaluated:

The **Least Significant Bit (LSB) size** is the amount by which the output changes in response to the input being incremented by 1.

$$LSB = \frac{FSR}{2^n - 1}$$

where FSR is the full scale range of the converter and n is the number of bits (resolution).

Offset Error is the difference between the nominal and actual offset points when the digital input is zero. The nominal offset is zero, so the offset is simply the current or voltage output measured when the digital input is zero. This error affects all codes by the same amount. It's expressed in LSB units or as a percentage of the full-scale range. The offset equals the value of the intercept calculated as part of the integral non-linearity (INL) measurement.

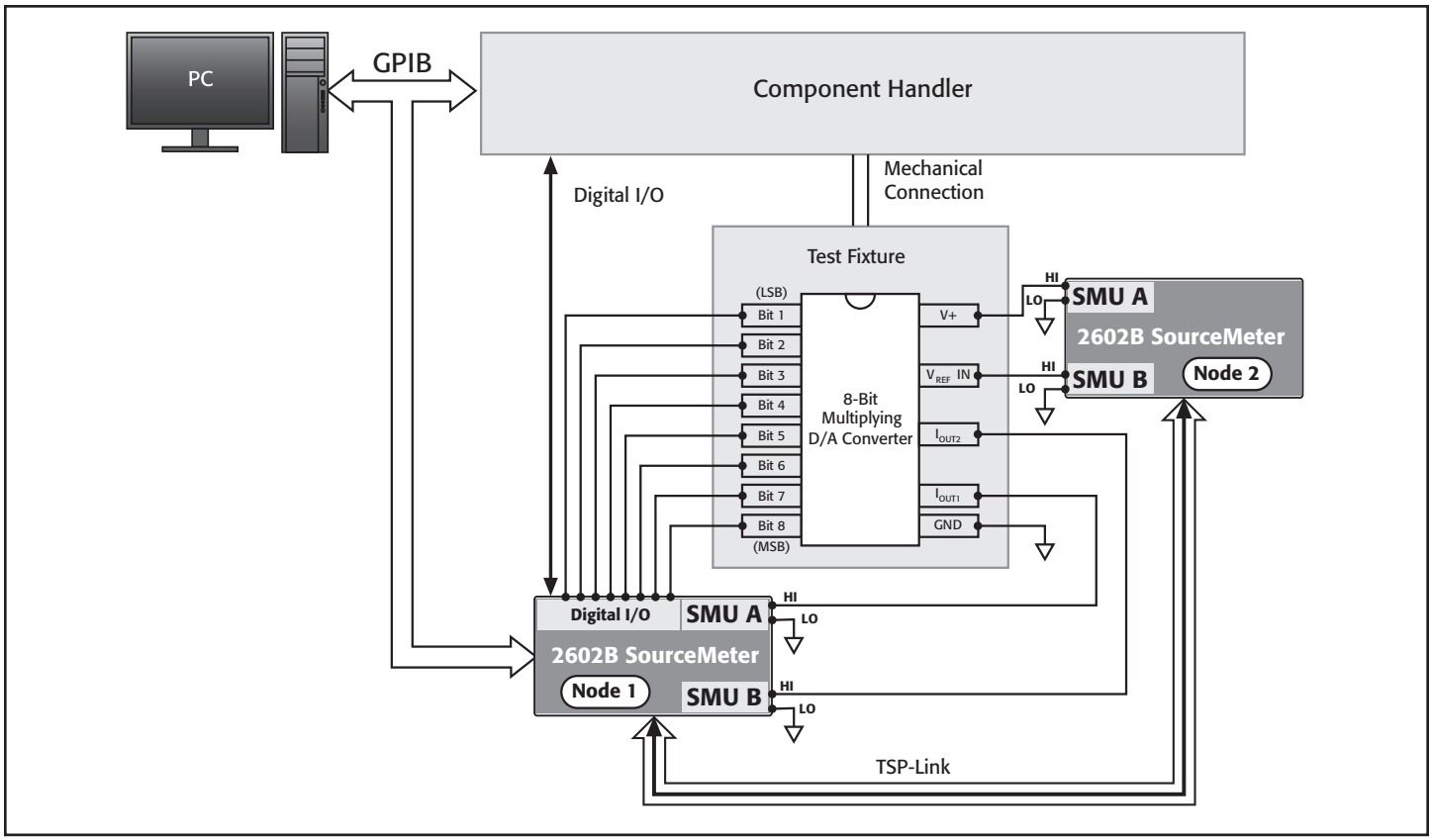

Figure 3. Setup using two Model 2602B SourceMeter instruments to test an 8-bit multiplying digital-to-analog converter

Gain Error is the difference between the nominal and actual gain points on the transfer function when the digital input is full scale and the offset error has been corrected to zero. This error represents a difference in the slope of the actual and ideal transfer functions and therefore corresponds to the same percentage error in each step. It's expressed in LSB units or as a percentage of the full-scale range.

Integral Nonlinearity (INL) describes the deviation of a DAC's actual transfer function from a straight line. For DACs, this deviation is measured at every step. The straight line can be either a best-fit straight line of the actual transfer function, which is determined so as to minimize these deviations, or a line drawn between the transfer function's end points. INL is expressed in LSB units or as a percentage of the full-scale range.

INL can be calculated in terms of LSB using the following equation:

$$INL = \frac{I_{CODE} - [(SLOPE * CODE) + INTERCEPT]}{LSB}$$

where CODE is the applied digital input, I_{CODE} is the output current (or voltage) measured at a given CODE, SLOPE is the slope of the reference straight line, and INTERCEPT is the "y-intercept" of the reference straight line. The resultant INL is expressed in units of LSB. When a best-fit reference line is used, the slope and intercept are calculated by linear regression. When an endpoint-based reference line is used, the slope is calculated just like LSB (FSR = $I_{MAXCODE} - I_{MINCODE}$) and has the same numeric value. The intercept in this case is the offset current (or voltage) of the converter measured when the digital input is zero, i.e. CODE =0.

Differential Nonlinearity (DNL) is the difference between an actual step height and the ideal value of 1LSB. Therefore, if the step height is exactly 1LSB, then the DNL error is zero. The target value for DNL (≤1LSB) ensures the DAC is monotonic. This means no data is lost because the output always changes in accord with the digital input; it increases in response to a digital increment, and decreases in response to a digital decrement. It's expressed in LSB units or as a percentage of the full-scale range.

DNL can be calculated in terms of LSB using this equation:

$$DNL = \frac{(I_{CODE} - I_{CODE-1})}{LSB} - 1$$

where I_{CODE} is the output current (or voltage) measured at a given CODE, and I_{CODE-1} is the current (or voltage) measured at the previous code.

For our DAC test example, the offsets, gain errors, and both INL and DNL will be determined for both outputs after the measurements are completed. For I_{OUT1}, the INL evaluation will be done using the best-fit straight-line method, and for I_{OUT2}, it will be done using the endpoint straight-line method.[1]

Test Script

Traditionally, a test engineer would create a test program, which resides on a PC or other controller. Such a program would likely consist of a test executive and a collection of function procedures and other subroutines. The executive controls the test flow by calling the various functions or subroutines in the proper order. The functions and subroutines do things such as send commands to the instruments in the test system to configure them and initiate measurements. They retrieve data and status information from the instruments. They process and evaluate the data and make pass/fail decisions about the device-under-test, and they may archive the data. In general, the controller will send the command sequences for the instruments over and over again as each DUT is tested, and it will continually retrieve data for evaluation. All of this communication between the controller and the instruments can significantly slow down a test. The Series 2600B Test Script Processor allows downloading a significant portion of the control program into volatile or nonvolatile SourceMeter memory. The program downloaded to the TSP is called a script. The script can be one long program, which performs multiple tests. However, by following good programming practice, it's possible to write a script that will create and call functions just like the control program in the PC. Once a function is created, it can be called from scripts and other functions residing in the Test Script Processor or from a test executive residing in the host controller. Since parameters can be passed to a function, they provide a simple way for the host controller to pass DUT-specific test parameters, such as input signal levels or inspection limits, to the SourceMeter test routine.

A well-documented sample script for testing a DAC is available for download on Keithley's web site, www.keithley.com. This script is fully functional and can be used with two Model 2602B SourceMeter instruments configured as shown in ***Figure 3***. The following code fragments are excerpts from the DAC test script and are provided to give the reader a feel for the new scripting language. Note that double dashes (--) indicate comments.

Let's look at a couple of typical commands:

```
node[1].smua.source.func = node[1].smua.OUTPUT _ DCVOLTS
node[1].smua.source.levelv = 0
```

The scripting language allows the use of aliases, which can make the code more readable and improve its execution speed. The following aliases have been defined for the DAC test example:

```
MASTER = node[1]       --Alias indicating control is via Node 1
SLAVE = node[2]        --Node 2 is controlled by MASTER via TSP-Link
IOUT1 = MASTER.smua    --Alias for SMU measuring current output #1
                       --IOUT1 is equivalent to node[1].smua
IOUT2 = MASTER.smub    --Alias for SMU measuring current output #2
                       --IOUT2 is equivalent to node[1].smub
DIO = MASTER.digio     --Alias for Digital I/O of 2602B #1
                       --DIO is equivalent to node[1].digio
VPLUS = SLAVE.smua     --Alias for SMU supplying V+ and measuring current draw
                       --VPLUS is equivalent to node[2].smua
VREF = SLAVE.smub      --Alias for SMU supplying reference voltage (Vref)
                       --VREF is equivalent to node[2].smub
```

The aliases are used throughout the example. Using the defined aliases, the example commands can be rewritten as:

```
IOUT1.source.func = IOUT1.OUTPUT_DCVOLTS
IOUT1.source.levelv = 0
```

In general, the scripting language doesn't require that variables be declared explicitly. They are declared and typed "on the fly," based on the values assigned to them. Exceptions are tables (i.e., arrays), which must be identified as such. All variables are global unless explicitly declared as local. The following "constants" appear in the following code fragments:

```
Vref = 10              --Use +10VDC reference voltage
IoutMax = 0.002        --Max expected current output
Nplc = 0.001           --Integration time for SMU A-to-D converters (in terms of power line cycles)
```

---

[1] The definitions in this application note draw from those outlined in the following documents:

- "Optimizing Setup Conditions for High Accuracy Measurements of the HI5741," Intersil Corporation, Milpitas, CA, Application Note 9619, May 1996.
- "Digital-Analog Converters Are a 'Bit' Analog," Maxim Integrated Products, Inc., Sunnyvale, CA, Application Note 1055, April 16, 2002.
- "Understanding Data Converters," Texas Instruments, Inc., Dallas, TX, Application Report SLAA013, 1995.

```
Nbits = 8              --Number of DAC control bits (digital inputs)
Ncodes = 2^Nbits       --Number of possible control codes
MaxCode = Ncodes - 1   --Decimal equivalent of full-scale code (255 for 8-bit DAC)
Lsb = Vref / MaxCode   --Nominal value of least significant bit
```

It is common practice to perform some initial setup of the instruments before beginning the actual test sequence. For our example, initial setup includes such things as setting up source functions and ranges, measurement functions and ranges, voltage sense modes and so on. All four SMUs are configured similarly. Only some of the setup commands for Node 1, SMU A follow:

```
MASTER.reset()                                  --Reset all Node 1 logical instruments to default settings
IOUT1.sense = IOUT1.SENSE_REMOTE                --Use REMOTE (4-wire) voltage sensing
IOUT1.source.func = IOUT1.OUTPUT_DCVOLTS        --Configure SMU to source DCV
IOUT1.source.rangev = 0                         --Set voltage source ranges;
                                                --2602B picks appropriate range based on programmed value
IOUT1.source.levelv = 0                         --To measure current, source zero volts on lowest range
IOUT1.source.limiti = 1.2 * IoutMax             --Set current compliance limit (20% over max)
IOUT1.measure.nplc = Nplc                       --Set integration times for all measurements
IOUT1.measure.autozero = IOUT1.AUTOZERO_AUTO    --Autozero for max accuracy;
IOUT1.measure.rangei = IoutMax                  --Set up current measurement range; Measurement
                                                --range for source function fixed at source range val
IOUT1.measure.filter.type = IOUT1.FILTER_REPEAT_AVG    --Use REPEAT filter
IOUT1.measure.filter.count = 5                  --Reading will be average of 5 consecutive measurements
IOUT1.measure.filter.enable = IOUT1.FILTER_ON   --Enable Node 1 SMU A digital filter
                                                --Set measurement parameters the 2602s will display
                                                  (if display is enabled)
                                                --Displays can be disabled to improve test speed
MASTER.display.screen = MASTER.display.SMUA_SMUB       --Digital port isn't affected by reset so user
                                                         must set desired initial state
DIO.writeport(0)                                --Set all digital control bits to zero
DIO.writeprotect(16128)                         --Write protect bits 9 through 14, which are reserved for
                                                --component handler control in this example.
```

The DAC test is performed after the initial setup is complete. Only the INL and DNL measurements made at the I_{OUT1} terminal are shown here. See the complete test script for the other tests. Note: A SourceMeter instrument always "assumes" it's measuring a current from its internal source. In this case, a POSITIVE current will flow *OUT* of its terminals and a NEGATIVE current will flow *IN* to them. As a result of this convention, a SourceMeter instrument operating in the "measure current only" mode, such as Node 1, SMUs A and B, will measure polarities opposite those that would be expected when using a typical ammeter. A positive current flowing from the circuit into the SourceMeter instrument will be measured as a negative current and vice versa.

```
IOUT1.source.output = IOUT1.OUTPUT_ON       --Turn ON SMU outputs
iout1 = {}  --Declare table to hold measurements at output IOUT1; table index begins with 1

for j = 0, MaxCode do --j is the code applied to the digital inputs
     DIO.writeport(j) --Apply digital inputs
     delay(0.001)     --Allow 1ms settling time
     iout1[j+1] = -IOUT1.measure.i() --Minus sign corrects for polarity of measurements
end --for

IOUT1.source.output = IOUT1.OUTPUT_OFF      --Turn OFF outputs of all SMUs
```

Once the measurements are completed, the Node 1 Test Script Processor performs all calculations and inspects the data for pass/fail status. The scripting language has an extensive math library, which allows the TSP to perform complex math calculations and eliminates the need to send data to the host controller for processing. The complete example program illustrates how the Model 2602B can perform a linear regression calculation.

```
--Compute maximum integral nonlinearity(INL)
--Check for monotonicity; Compute maximum differential nonlinearity(DNL)
--Slope_bf and intercept_bf are the slope and intercept of the best-fit straight line
    inlmax_iout1 = 0
    dnlmax_iout1 = 0
    mono_iout1 = "Monotonic"
```

```
for j = 0, MaxCode do
    inl_iout1 = math.abs(iout1[j+1]-(slope_bf * j + intercept_bf))  --Calcs for IOUT1
    if inl_iout1 > inlmax_iout1 then
        inlmax_iout1 = inl_iout1
    end --if

    if j > 0 then
        --Test for monotonicity
        diff_iout1 = iout1[j] - iout1[j-1]
        if diff_iout1 < 0 then
            mono_iout1 = "NON-Monotonic"
        end --if
        --Compute dnl and test for max.
        dnl_iout1 = math.abs(diff_iout1 - Lsb)
        if dnl_iout1 > dnlmax_iout1 then
            dnlmax_iout1 = dnl_iout1
        end -if
    end --if
end --for

inl_iout1_lsb = inlmax_iout1 / Lsb     --Express INL and DNL in terms of nominal LSB
dnl_iout1_lsb = dnlmax_iout1 / Lsb
```

Once the various DAC parameters are calculated, the TSP inspects the values and determines the pass/fail status of the part. It then sends the appropriate binning command to the component handler by writing a digital bit pattern to the Node 1 DIO port.

```
if PartStatus ="GOOD" then
    DIO.writeport(GoodBitPattern)  --Send "good part" bit pattern to component handler
else
    DIO.writeport(BadBitPattern)  --Send "bad part" bit pattern to component handler
end --if
```

Since all of the test data was processed and evaluated by the TSP, it is not necessary to send any data to the host controller. However, when data retrieval is required for SPC or to satisfy other data logging or record-keeping requirements, it is easily accomplished. The "print" function writes the specified parameters to the 2602B output queue where the host controller can then upload them. If desired, data and/or test results can be shown on the instrument front panel displays. Standard "C" formatting strings can be used to format the data.

```
--Send the monotonicity results and max INL and DNL values measured at IOUT1
print(string.format("%s, %1.2f, %1.2f", mono_iout1, dnl_iout1_lsb, inl_iout1_lsb))
--Display INL & DNL on front panel displays
MASTER.display.clear()
MASTER.display.setcursor(1,1,0)
MASTER.display.settext(string.format("INL= %1.2f LSBs", inl_iout1_lsb))
MASTER.display.setcursor(2,1,0)
MASTER.display.settext(string.format("DNL= %1.2f LSBs, dnl_iout1_lsb))
```

Downloading and Running the Script and Retrieving Data

Now that we've offered an overview of what a script looks like, let's discuss briefly how to create it, download it, and run it.

Since the DAC example uses two SourceMeter instruments connected together by TSP-Link, the first thing to do is set up TSP-Link. The TSP-Link cable is a readily available Ethernet crossover cable. Cat 6 cable is recommended because of its higher bandwidth capability, but both Cat 5e and Cat 6 cables will work. As previously stated, each instrument connected by TSP-Link is assigned a unique Node Number, similar to a GPIB address. The node number (from 1 and 32) can be set from the front panel or via remote control. To run the DAC test example, set the master node number to 1 and the slave node number to 2 (the master is the SourceMeter instrument connected to the host controller). Before it's possible to communicate via TSP-Link, the bus must be initialized or reset. This, too, can be done from the front panel or via remote control. Do the following to reset TSP-Link remotely:

- To check the status of TSP-Link, send "print(tsplink.state)" and enter the response, which will be either "online" or "offline."
- If TSP-Link is off line, then you reset it (bring it on line) by sending "tsplink.reset()."
- Either the host controller or the "master" TSP can reset TSP-Link.

A host controller can send individual commands to a Series 2600B SourceMeter instrument and interact with it like any typical programmable instrument, but the power of the Series 2600B clearly lies in its ability to accept and run complete test scripts. Obviously, a script can be much more than just a list of instrument commands; it can be a complete program, just like one that can be created using any of the common programming languages. Although scripts can be created using any text editor, Keithley provides an application called Test Script Builder, which can be used to create, organize, and debug scripts. It can download scripts to either volatile or nonvolatile memory and then it can run them. It's also possible to load and run scripts using applications created in other languages, such as Visual Basic, Visual C/C++, or LabVIEW. Once the script is in instrument memory, it can even be run from the front panel. When the example DAC test script is loaded and run, it creates all of the various functions required for the DAC test. The functions include one named TestDAC, which serves as a TSP-resident test executive that calls various other functions to perform the complete DAC test. The host controller only has to send "TestDAC()" to the master 2602B to perform the complete test.

The Model 2602B can eliminate the need to send data to the host controller for processing. However, simple "print" commands make it possible to retrieve data when it is required. Each print statement requires a corresponding "enter" statement in the host program, to pull the data from the output queue.

Equipment List

The following equipment is required to assemble a DAC test system and run the example test script available from the Keithley web site, www.keithley.com

1. Two Keithley Model 2602B Dual Channel SourceMeter® instruments.
2. PC with KPCI-488LPA GPIB Interface Card or KUSB-488B GPIB Interface Adapter or equivalent.
3. Keithley 7007 IEEE-488 interface cable.
4. CAT 5e Ethernet crossover cable to connect SourceMeter instruments via TSP-Link.
5. AD7523JN 8-Bit Multiplying DAC (script was verified using this DAC model).
6. Component handler with test fixture (part of production test system; not required to simply run script).
7. Custom DB-25 digital I/O handler interface cable to interface the instrument di gital I/O to the handler (part of production test system; not required to simply run script).
8. Test leads to connect the instrument to the test fixture.

Test System Safety

Many electrical test systems or instruments are capable of measuring or sourcing hazardous voltage and power levels. It is also possible, under single fault conditions (e.g., a programming error or an instrument failure), to output hazardous levels even when the system indicates no hazard is present. These high voltage and power levels make it essential to protect operators from any of these hazards at all times. Protection methods include:

- Design test fixtures to prevent operator contact with any hazardous circuit.
- Make sure the device under test is fully enclosed to protect the operator from any flying debris. For example, capacitors and semiconductor devices can explode if too much voltage or power is applied.
- Double insulate all electrical connections that an operator could touch. Double insulation ensures the operator is still protected, even if one insulation layer fails.
- Use high-reliability, fail-safe interlock switches to disconnect power sources when a test fixture cover is opened.
- Where possible, use automated handlers so operators do not require access to the inside of the test fixture or have a need to open guards.
- Provide proper training to all users of the system so they understand all potential hazards and know how to protect themselves from injury.

It is the responsibility of the test system designers, integrators, and installers to make sure operator and maintenance personnel protection is in place and effective.

KEITHLEY
A Tektronix Company

A Greater Measure of Confidence

KEITHLEY INSTRUMENTS, INC. ■ 28775 AURORA RD. ■ CLEVELAND, OH 44139-1891 ■ 440-248-0400 ■ Fax: 440-248-6168 ■ 1-888-KEITHLEY ■ **www.keithley.com**

BRAZIL
55-11-4058-0229
www.keithley.com

CHINA
86-10-8447-5556
www.keithley.com.cn

FRANCE
01-69868360
www.keithley.fr

GERMANY
49-89-84930740
www.keithley.de

INDIA
080-30792600
www.keithley.in

ITALY
02-5538421
www.keithley.it

JAPAN
Tokyo: 81-3-6714-30
Osaka: 81-06-6396-1630
www.keithley.jp

KOREA
82-2-6917-5000
www.keithley.co.kr

MALAYSIA
60-4-643-9679
www.keithley.com

MEXICO
52-55-5424-7905
www.keithley.com

SINGAPORE
01-800-8255-2835
www.keithley.com.sg

SWITZERLAND
41-56-460-78-90
www.keithley.ch

TAIWAN
886-3-572-9077
www.keithley.com.tw

UNITED KINGDOM
044-1344-392450
www.keithley.co.ukw

Printed in the U.S.A

No. 2614

2.20.13