KEITHLEY   A GREATER MEASURE OF CONFIDENCE

Using the Wafer Map Parameters Option with Cascade Nucleus Prober Software and the Model 4200-SCS Semiconductor Characterization System

Introduction

Parameter Wafer Mapping allows users to create a visual display of test data, showing parameters as different colored dies on a wafer map. This Technical Note describes the use of this feature on Cascade Microtech's Summit Series semi-automatic wafer probers, driven by the Nucleus™ software, with electrical measurements taken by the Model 4200-SCS Semiconductor Characterization System. This note assumes the reader is already familiar with the principles and operation of the Summit prober, Nucleus software, and the Model 4200-SCS's KTEI software environment. When using different hardware or software, the steps involved in implementing this feature will vary, depending on the model or version chosen.

Overview

The Model 4200-SCS controls the Summit prober via a GPIB bus, as shown in *Figure 1*.

The GPIB receiving card is a part of the Cascade 12000 Probe Station PC (or simply *Prober PC*), which also contains a stepper motor control card to send motion commands to the probe station. Nucleus UI, the software running on the Prober PC, allows users to create and maintain wafer maps, virtual models of silicon wafers, which contain exact information about wafer, die, and in-die geometry. When the external controller (Model 4200-SCS, in this case) issues a command such as "Move to Next Die," Nucleus UI translates this command into an instruction for the prober's stepper motors to move the chuck by a certain distance in real units (millimeters or inches) in the X-Y plane. As an example of a two-way communication between a controller and the Prober PC, the Model 4200-SCS can request the current die coordinates from Nucleus UI, and upon receiving them, add them to the set of data acquired on that die location.

Parameter Wafer Mapping takes advantage of such two-way communications capabilities between the controller and the prober. After completing an electrical measurement on a die structure, the Model 4200-SCS can send the value of the measured (or calculated/extracted) parameter, along with the die coordinates, to Nucleus UI's pre-configured wafer map, which then marks the die with the proper color and intensity (hue) based on the value of the parameter. When wafer testing is complete, the user can visually evaluate parameter distribution across the wafer, looking for patterns indicative of the wafer processing equipment or processes (*Figure 2*).
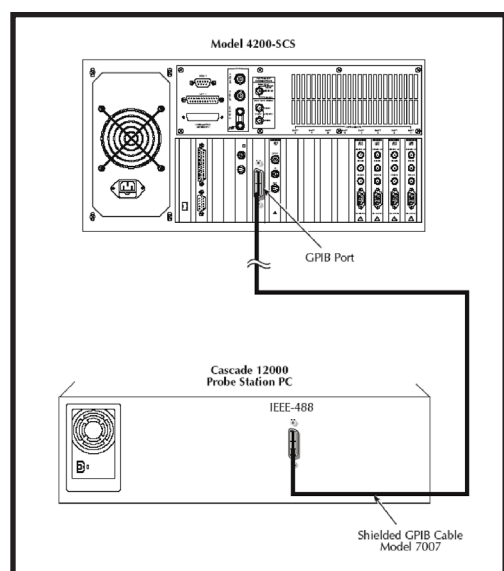

Figure 1.

Using the Wafer Map Parameters Option with Cascade Nucleus Prober Software and the Model 4200-SCS Semiconductor Characterization System
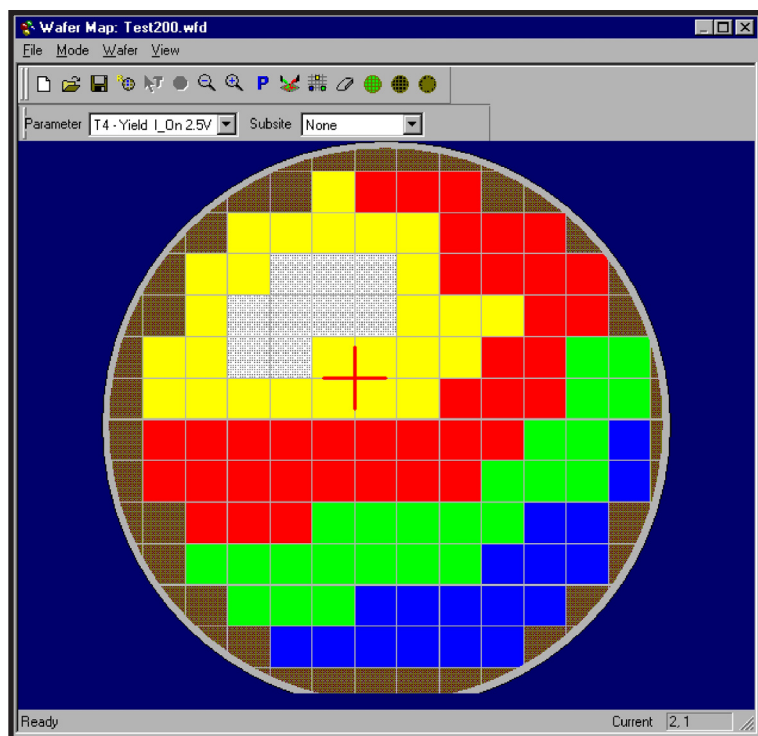

Figure 2.

Preparation and Setup

Before the Parameter Wafer Mapping feature can be used, the user must set up both the hardware and software of the controller and the prober. This process is described and illustrated in great detail in Appendix J—"Cascade Summit-12000 Prober" of the Model 4200-SCS's Reference Manual. Using the Manual as a guide, perform the following steps:

1. Boot up and log into the Cascade Prober PC; launch Nucleus UI.

2. Connect the Model 4200-SCS to the Prober PC via GPIB cable.

3. Configure and enable GPIB communications in Nucleus UI.

4. Set up the wafer map geometry using the Wafer Map Wizard in Nucleus UI.

5. Create a site definition and probe list in Nucleus UI.

6. Load, align, and contact the physical wafer on the Cascade 12000 prober.

7. Verify/modify the prober configuration file ***prbconfig_CC12k**.dat* in the Model 4200-SCS.

8. Add the Cascade 12000 Prober to the system configuration in KCON (4200-SCS).

9. Create a KITE project or use an existing Probesites project in KITE (4200-SCS).

If all of the above steps are performed correctly and successfully, the system should be ready to go. At this point, it may be a good idea to verify prober control operation by running the KITE project or parts of it. Usually, three or four consecutive qualifying electrical measurements are sufficient indicators of good prober contact, correct wafer map configuration and wafer alignment, and consistent controller-prober communication.

Configuring the Wafer Map Parameters

The process of setting up the Wafer Map Parameters is best described in the section of the Nucleus Manual titled *Wafer Map Parameters/Using the Parameter Dialog.* This section outlines the following minimum steps:

1. Open the *Wafer Map Data Parameter Setup* dialog from the wafer map menu (***Figure 3***).

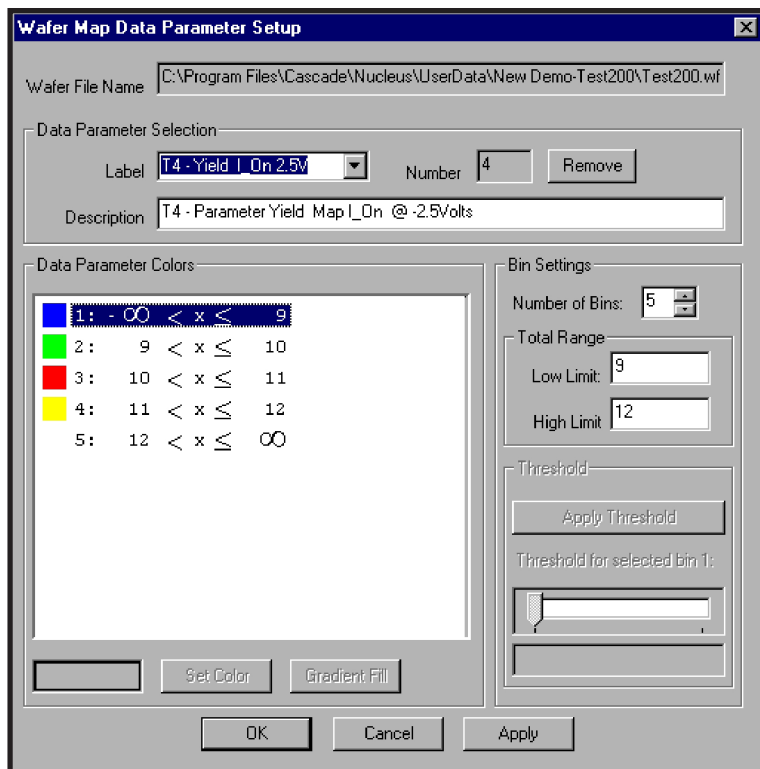2. In the Data Label field, enter a name for your test parameters.
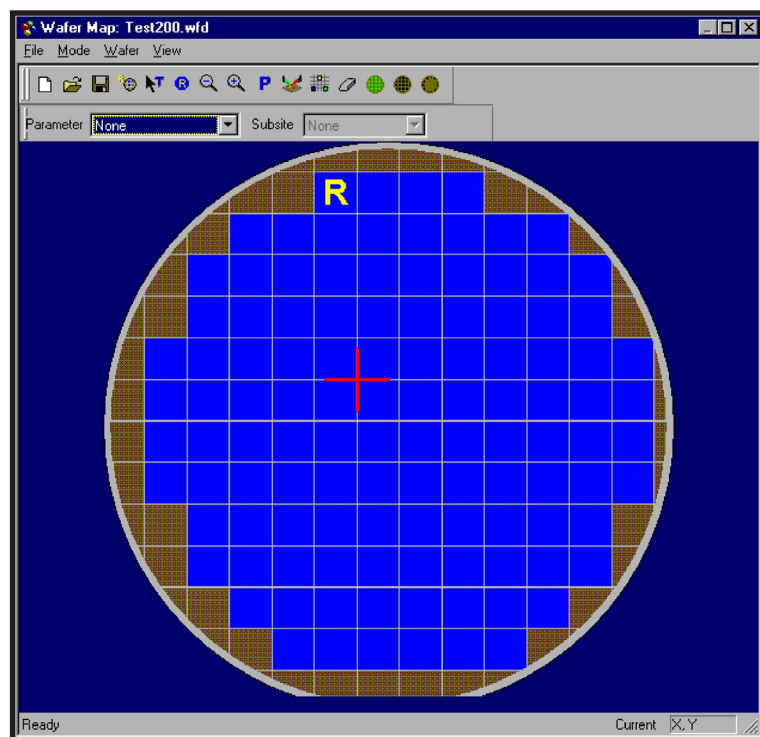
Figure 3.

Figure 4.

3. In the Parameter Description field, describe the test parameters.

4. Select the number of sorting bins.

5. Fill in the parameter range for bins.

6. Optionally, the user may set up various color options.

The wafer map is now ready to receive parameters from the controller (the Model 4200-SCS), as shown in *Figure 4*. These parameters may only be sent via remote GPIB commands; there is no user interface function for setting test result values.

Acquiring and Communicating the Wafer Map Parameters

Before a parameter can be sent to the wafer map, it must first be obtained, whether by direct electrical measurement or as a product of a subsequent calculation. For example, in order to send parameter R (resistance) to the wafer map, the Model 4200-SCS must first measure voltage (V) and current (I), and then calculate R as a product of the division of V by I.

Two other pieces of information are required to communicate the parameter to Nucleus using the *:PROBeplan:MARK:XYDie:VALue* command. One of them is a set of current die coordinates (X, Y count relative to a designated Home Die). The coordinates can be obtained by sending a query to Nucleus via the *:MOVe:PROBeplan:ABSolute:DIE?* command. Also, Nucleus needs the parameter number, since more than one parameter can be configured on the wafer map.

So, the steps are:

1. Move prober chuck to the current die position.

2. Perform an electrical measurement using the Model 4200-SCS.

3. (Optional) Calculate a parameter from the measurement results.

4. Send a query to Nucleus about current die X,Y coordinates.

Using the Wafer Map Parameters Option with Cascade Nucleus Prober Software and the Model 4200-SCS Semiconductor Characterization System

5. Receive coordinates, decode the string (extract X,Y).

6. Send the parameter mapping command using the X,Y coordinates.

7. Repeat steps 1–6 until the entire wafer is tested.

Steps 1 and 7 can be accomplished by calling the *PrMovNxt* user module (*PrbGen* library) in KITE project, and setting the project to execute multiple sites, corresponding to the number of die to be tested on a wafer. Steps 2 through 6 can be accomplished using a custom user module.

The source code sample shown in Table 1 illustrates the implementation of this sequence in the Model 4200-SCS's KTEI programming environment. To simplify the task, a user can start from the existing KITE project called *ProbeSites* (installed standard on every Model 4200-SCS under the C:\S4200\kiuser\Projects\... directory). This project can then be modified by adding a new UTM (User Test Module). A UTM points to a pre-compiled piece of C code (User Module), written using the KULT interface.

Table 1

```
#includee "keithley.h"
#include "ieee_32.h"

int Main_module( double volt, int GPIBaddr, double *curr )
{
/* USRLIB MODULE CODE */
// Variable declaration:
    char CommandString[99], RawReading[99];         // Internal strings for GPIB communication
    int     rcv_size;                               // Actual # chars read from instrument
    int     max_size=99;                            // Max # of characters to read from instrument
    int temp1, temp2, X, Y;                         // X,Y die coordinates
    int Param;
    long status;
    double R;                                       // Declare calculated parameter R

forcev(SMU1, volt);                                 // Force voltage V on SMU1
measi(SMU1, curr);                                  // Measure currnt I on SMU1
R=volt/(*curr);                                     // Calculate R

sprintf(CommandString, ":mov:prob:abs:die? 2");     //Request current die X,Y cordinates
send(GPIBaddr,CommandString,&status);               //Send string to the prober

enter(RawReading, max_size, &rcv_size, GPIBaddr, &status);  //Read raw string from the prober
if (sscanf(RawReading, "%d %d", &temp1, &temp2) != 2)       //Extract X,Y coordinates from raw string
    {
        return(ERROR_PARSING);                      // Error parsing the reading
    }

    X = temp1;
    Y = temp2;
Param = 1;

sprintf(CommandString, ":prob:mark:xydie:val %d %d %d %g",X,Y,Param,R); //Compose GPIB string
send(GPIBaddr,CommandString,&status);                                   //Send string to the prober
enter(RawReading, max_size, &rcv_size, GPIBaddr, &status);              //Read response from the prober

return (OK);

/* USRLIB MODULE END   */
}               /* End Main_module.c */
```

In the code sample in **Table 1**, current and voltage measurements, parameter calculation (R = V/I), and prober communication all take place within the same piece of code. This was done intentionally to demonstrate the entire concept of parameter mapping.

However, it's better programming practice to use the parameter mapping module as a function that's called from within the main code. Only three arguments need to be passed to this function: GPIB address of the prober, parameter number, and parameter value. Such implementation would allow simplifying the code in **Table 1** to that shown in **Table 2**.

Table 2

```
#include "keithley.h"

int Main_module_compact( double volt, int GPIBaddr, double *curr )
{
/* USRLIB MODULE CODE */
double R;                    // Declare calculated parameter R

forcev(SMU1, volt);          // Force voltage V on SMU1
measi(SMU1, curr);           // Measure currnt I on SMU1
R=volt/(*curr);              // Calculate R

Map_parameter(28, 1, R);     //Call function: Map_parameter(GPIB_Address,Param_code,Param_value)

return (OK);
/* USRLIB MODULE END  */
}               /* End Main_module_compact.c */
```

It's important to emphasize here that the code samples shown in **Table 1** or **Table 2** are only suggested implementations of this feature. Users are welcome and encouraged to modify this code to suit their specific needs, applications, and personal programming style and preferences.

Conclusion

The parameter wafer mapping feature enhances the functionality of the Model 4200-SCS, making it a powerful, versatile, low cost solution for low volume semiconductor wafer testing.

**Using the Wafer Map Parameters Option with
Cascade Nucleus Prober Software and the
Model 4200-SCS Semiconductor Characterization System**

KEITHLEY