During the first compile pass on gcc960, data flow analysis provides information on how procedures interface with one another. This data is used for two purposes: to optimize code organization for better instruction cache hit ratios, and to recognize the best opportunities for inlining procedures.

4.5 Branch Prediction

The run time profile can optionally provide heuristic data back to a branch prediction post processor. This data provides results indicating which path was usually taken for each CTRL instruction encountered during execution. This data is then used to set the appropriate bit in the CTRL instruction opcode. This optimization provides an average of 3-5% performance improvement to applications.

5. SUMMARY

The i960 processor family has several members covering a wide range of performance. The i960 offers an easy to use, robust architecture, from which both application designers and tool designer can create powerful applications.
Inside Symbolic Debugging for the M68HC16 Microprocessors

Walter Banks, President
Byte Craft Limited
421 King Street North
Waterloo, Ontario
Canada N2J 4E4
Phone (519) 888-6911
FAX (519) 746-6751

Abstract:
The Motorola M68HC16 Microprocessor has a background mode serial port to facilitate accessing the processor during development or as access for production line testing or regular maintenance. Byte Craft Limited developed a Host Symbolic Debugging package during development of a C compiler for the M68HC16 which uses the background mode serial port.

This talk covers two aspects of Symbolic Debugging on the M68HC16. The first part covers the hardware and software design of the interface to the background mode port. The second part covers the design of the host software and how it can be used for application software development, and product testing.

Background
For the last few years Byte Craft Limited has been using a common user interface for number of in house developments. This interface is based on a LISP-like command language called ASK that is designed to quickly build up user queries to a target. ASK was originally designed as the query language to an object oriented simulator. By replacing the simulator behavior model with an emulator, or a data link to a production target a very powerful user front end was generated.

Byte Craft started adding simple monitors to our embedded applications as early as 1983. [3] These monitors provided a minimum of user commands and required a smart host to utilize them. The ASK-based front end was mated to the embedded monitors achieving significant enhancements to the useability of the embedded monitors for uses other than bugs detection and repair. With easily generated host software it became easy to tailor a host piece for use by semi-skilled production line worker and using the same embedded code and a different host package to do full symbolic debugging for product fault finding.

The background mode port on the 6816 family further enhanced our ability to access a target processor by reducing the run time overhead to essentially nothing. The background mode port can access a target without any specialized application software allowing even more freedom to both access and understand the inner workings of an application.

Design objectives
The design objectives for production line testing and debugging include the following objectives. The total system has to have little effect on the overall production cost of the system. The combination of background port and host support software must be able to functionally replace existing application specific test code. Beyond these basic ground rules we had the following objectives.

1. Minimum code embedded within a product to facilitate standard production testing. ROM space is always a problem in any embedded product. In an application product test code is pure overhead and should be minimized.

2. Host support code that offers both flexibility and use by nontechnical personnel. The host platform is an IBM-PC or compatible interfaced to the target embedded system.

3. Facility for debugging the product with access to the current state information and symbolic debugging host support. To be useful in debugging complex systems, a host package should be able to access the system under test symbolically and have rapid user access to key program listings.

4. Ability to display extensive diagnostic information on request. User flexibility is gained by organizing the code to answer users' requests for information rather than displaying anticipated useful data.

5. Complete host control of the system under test. The host software is capable of resetting and controlling the target system. This single point of control allows clear,
Inside Symbolic Debugging for the M68HC16 Microprocessors

Walter Banks, President
Byte Craft Limited
421 King Street North
Waterloo, Ontario
Canada N2J 4E4
Phone (519) 888-6911
FAX (519) 746-6751

Abstract:
The Motorola M68HC16 Microprocessor has a background mode serial port to facilitate accessing the processor during development or as access for production line testing or regular maintenance. Byte Craft Limited developed a Host Symbolic Debugging package during development of a C compiler for the M68HC16 which uses the background mode serial port.

This talk covers two aspects of Symbolic Debugging on the M68HC16. The first part covers the hardware and software design of the interface to the background mode port. The second part covers the design of the host software and how it can be used for application software development, and product testing.

Background
For the last few years Byte Craft Limited has been using a common user interface for number of in house developments. This interface is based on a LISP-like command language called ASK that is designed to quickly build up user queries to a target. ASK was originally designed as the query language to an object oriented simulator. By replacing the simulator behavior model with an emulator, or a data link to a production target a very powerful user front end was generated.

Byte Craft started adding simple monitors to our embedded applications as early as 1983. [3] These monitors provided a minimum of user commands and required a smart host to utilize them. The ASK-based front end was mated to the embedded monitors achieving significant enhancements to the useability of the embedded monitors for uses other than bugs detection and repair. With easily generated host software it became easy to tailor a host piece for use by semi-skilled production line worker and using the same embedded code and a different host package to do full symbolic debugging for product fault finding.

The background mode port on the 6816 family further enhanced our ability to access a target processor by reducing the run time overhead to essentially nothing. The background mode port can access a target without any specialized application software allowing even more freedom to both access and understand the inner workings of an application.

Design objectives
The design objectives for production line testing and debugging include the following objectives. The total system has to have little effect on the overall production cost of the system. The combination of background port and host support software must be able to functionally replace existing application specific test code. Beyond these basic ground rules we had the following objectives.

1. Minimum code embedded within a product to facilitate standard production testing. ROM space is always a problem in any embedded product. In an application product test code is pure overhead and should be minimized.
2. Host support code that offers both flexibility and use by nontechnical personnel. The host platform is an IBM-PC or compatible interfaced to the target embedded system.
3. Facility for debugging the product with access to the current state information and symbolic debugging host support. To be useful in debugging complex systems, a host package should be able to access the system under test symbolically and have rapid user access to key program listings.
4. Ability to display extensive diagnostic information on request. User flexibility is gained by organizing the code to answer users' requests for information rather than displaying anticipated useful data.
5. Complete host control of the system under test. The host software is capable of resetting and controlling the target system. This single point of control allows clear,
unambiguous access to the system under test. This feature is desirable for developers and essential for production personnel using the system.

6. Interface to the embedded system should be simple and low in cost.

Hardware

The M68HC16 Background Mode port allows an external interface to cause the target processor to reset, sync breakpoints to a specific address and to transfer data over a pair of synchronous bit serial ports. Data transfers are accomplished similar to the SPI bus architecture of the serial busses found on many low end embedded systems.

Each data transaction has two components. A command is sent to the M68HC16 and is followed by either arguments to the M68HC16 or responses from the M68HC16. The command set allows read and write access to all registers, and memory in the system as well as the ability to initialize execution.

The MC68HC16 can be in either run mode or background mode. The background mode is a special state of the processor where normal execution is suspended and communication through the background mode serial port is enabled. This mode may be initialized by either the execution of a special background mode instruction or by the assertion of an external breakpoint interrupt. A one time decision is made by the processor at reset time to decide if background mode will be allowed at all. This is needed to lock out spurious breakpoint interrupts and to allow the software to ignore the soft background mode instructions BGND.

Motorola strongly suggests that a background mode port be part of any embedded design using the 6816 part. This requires an eight pin connector arranged as two rows of four bits separated by 0.1 inch spacing. The connector is electrically identical to the background mode connector suggested for use on the 68332 processors. For the product in production this connector will add about a dollar to production cost. Overall engineering cost is small.

Software

Production line testing is often done by personnel with limited technical training in the specific application technology. In production lines involving volume production, it is useful to lead the test personnel through
unambiguous access to the system under test. This feature is desirable for developers and essential for production personnel using the system.

6. Interface to the embedded system should be simple and low in cost.

Hardware

The M68HC16 Background Mode port allows an external interface to cause the target processor to reset, sync breakpoints to a specific address and to transfer data over a pair of synchronous bit serial ports. Data transfers are accomplished similar to the SPI bus architecture of the serial busses found on many low end embedded systems.

Each data transaction has two components. A command is sent to the M68HC16 and is followed be either arguments to the M68HC16 or responses from the M68HC16. The command set allows read and write access to all registers, and memory in the system as well as the ability to initialize execution.

The MC68HC16 can be in either run mode or background mode. The background mode is a special state of the processor where normal execution is suspended and communication through the background mode serial port is enabled. This mode may be initialized by either the execution of a special background mode instruction or by the assertion of an external breakpoint interrupt. A one time decision is made by the processor at reset time to decide if background mode will be allowed at all. This is needed to lock out spurious breakpoint interrupts and to allow the software to ignore the soft background mode instructions BGND.

Motorola strongly suggests that a background mode port be part of any embedded design using the 6816 part. This requires an eight pin connector arranged as two rows of four bits separated by 0.1 inch spacing. The connector is electrically identical to the background mode connector suggested for use on the 68332 processors. For the product in production this connector will add about a dollar to production cost. Overall engineering cost is small.

Software

Production line testing is often done by personnel with limited technical training in the specific application technology. In production lines involving volume production, it is useful to lead the test personnel through
a number of simple steps. With the right host support software Quality Assurance (QA) data can be automatically recorded un-obtrusively for further analysis. Quality control experts agree that the single largest cause of error is the production line test personnel either measuring or recording the wrong values [6]. The host support software required to support production line test personnel has to have intimate knowledge of the the application code so that when changes are made to the application, the production line test code is able to reference key values symbolically. The production line host support code needs to be able to be easily modified to reflect the operational needs of the production line. Often the production line tests need to be revised and altered to fully meet the needs of the target system. Finally the production line host support software needs to record key information on the target system both as a function of Quality Assurance needs and provide information that will be of use to the system designers in evaluating both design decisions and design assumptions.

The second user is the design or production engineer brought in to debug a production problem. The same key information is needed by the host support package. In this mode, the host package is mostly a Source Debugger [7] with the ability to respond to questions asked by the engineer involved. At the target end, the same internal code is executing but the host, rather than controlling the production line person now is subservient to the needs of the engineer in charge.

The Background Mode provides all that is essential to the development engineer. Production line testing can be enhanced by seeding the target code with predetermined breakpoints. The C6816 compiler is enhanced to assist in the generation of production line test code in the following two ways.

1. The compiler looks for a void function with void return called _STARTUP. This function is called directly on reset before most of the normal initialization code. Variables have been preserved and very little of the processor state has been disturbed. This feature is a standard in all of our high level language code generation products.

2. A predefined inline function BGND() will generate an enter background mode instruction in the compiler generated code. If the processor is running with the background mode disabled then the BGND() will cause the processor to execute an illegal instruction trap.

Summary

The problems related to production line testing are being reduced by the co-operation of microprocessor vendors. This paper identifies our solutions to the problem. We have seen the need to be flexible and provide solutions to the production line test problems in an effective and timely manner.

References


4. Scott Howard, "BD32 - Background mode Debugger" for the Motorola 68300 series of microcontrollers. Motorola BBS. Motorola Semiconductor Products, Canada Vancouver, B.C., Canada June 1991

5. P&E Microcomputer Systems, Inc. P.O. Box 2044, Woburn MA 01888-2044. (617) 944-7585

a number of simple steps. With the right host support software Quality Assurance (QA) data can be automatically recorded un-obtrusively for further analysis. Quality control experts agree that the single largest cause of error is the production line test personnel either measuring or recording the wrong values [6]. The host support software required to support production line test personnel has to have intimate knowledge of the application code so that when changes are made to the application, the production line test code is able to reference key values symbolically. The production line host support code needs to be able to be easily modified to reflect the operational needs of the production line. Often the production line tests need to be revised and altered to fully meet the needs of the target system. Finally the production line host support software needs to record key information on the target system both as a function of Quality Assurance needs and provide information that will be of use to the system designers in evaluating both design decisions and design assumptions.

The second user is the design or production engineer brought in to debug a production problem. The same key information is needed by the host support package. In this mode, the host package is mostly a Source Debugger [7] with the ability to respond to questions asked by the engineer involved. At the target end, the same internal code is executing but the host, rather than controlling the production line person now is subservient to the needs of the engineer in charge.

We have been implementing our host support software using a macro language called ASK. This package is a general purpose string processing macro language [1,2,3] that has been found to be a powerful solution to mating the information from our compilers and the users that need that information. As a general purpose interface, ASK can communicate with equal ease with simulators, emulators, on board monitors and background mode ports. Our host support application code is short. Our symbolic debugger is less than 100 lines and production line test code is often of similar length. This allows easy changes and more importantly ease of maintenance and understanding.

The Background Mode provides all that is essential to the development engineer. Production line testing can be enhanced by seeding the target code with some predetermined breakpoints. The C6816 compiler is enhanced to assist in the generation of production line test code in the following two ways.

1. The compiler looks for a void function with void return called _STARTUP. This function is called directly on reset before most of the normal initialization code. Variables have been preserved and very little of the processor state has been disturbed. This feature is a standard in all of our high level language code generation products.

2. A predefined inline function BGND() will generate an enter background mode instruction in the compiler generated code. If the processor is running with the background mode disabled then the BGND() will cause the processor to execute an illegal instruction trap.

Summary

The problems related to production line testing are being reduced by the co-operation of microprocessor vendors. This paper identifies our solutions to the problem. We have seen the need to be flexible and provide solutions to the production line test problems in an effective and timely manner.

References

4. Scott Howard, "BD32 - Background mode Debugger" for the Motorola 68300 series of microcontrollers. Motorola BBS. Motorola Semiconductor Products, Canada Vancouver, B.C., Canada June 1991
5. P&E Microcomputer Systems, Inc. P.O. Box 2044, Woburn MA 01888-2044. (617) 944-7585