HARRIS REAL TIME EXPRESS:
A NEW CONCEPT FOR REALTIME CONTROL

AGENDA - ROCHESTER RTX SEMINAR

- Introduction/ Technology Overview  - Auditorium
- RTX 2000 Architecture  - Auditorium
- RTX Applications & Interfacing  - Auditorium
- RTXDS Background  - Auditorium
- Lunch  - Auditorium
- Guest Speakers  - Auditorium
- RTXDB  - May Room
- Forth Compiler & Utilities  - May Room
- Debugger  - May Room
RTX 2000 Architecture

- Based on Novix 4016/6016 Architecture
- On-chip parameter and return stacks
- On-chip peripherals
  - Multiplier
  - Interrupt controller
  - Counter/timers
- Address 1 megabyte of memory
Architecture - 2

SEMICONDUCTOR PRODUCTS DIVISION

RTX 2000 Block Diagram

STACKS

- PARAMETER STACK
  - 256 WORDS DEEP
  - 16 BITS WIDE
  - Top two elements are available through on-chip registers

- RETURN STACK
  - 256 WORDS DEEP
  - 21 BITS WIDE
    - 15 BITS PROGRAM COUNTER
    - 4 BITS CODE PAGE
    - 1 BIT INTERRUPT STATUS
    - 1 BIT DATA PAGE FLAG
  - Top element available through on-chip register
Registers

- Top - top element of Parameter stack
- Next - second element of Parameter stack
- Index - top element of Return stack, loop counter
- Control/Status Configuration Register (CR)
- Interrupt Base/Control Register (IBC)
- Multistep Divide (MD) - used for step math or general purpose
- Square Root (SR) - used for square roots or general purpose
- Memory Page registers
  - Code Page - instruction fetches
  - Data Page - memory access (I/O, etc.)
  - User Page - user memory access
- I/O Devices
  - Multiplier output
  - Counters/Timers
  - Interrupt mask
  - Interrupt vector
  - Stack pointers
  - Stack limits

On-Chip Peripherals

- Stack Controllers
  - Programmable limit registers
  - Generate interrupts on overflow and underflow conditions
- Interrupt Controller
  - 14 interrupt sources, internal and external
  - 13 maskable, 1 non-maskable
  - Software interrupt
- Multiplier
  - Single cycle 16 x 16 multiply with 32-bit result
  - Forth * in 3 cycles
- Counter/Timers (3)
  - 16-bit down counters
  - Clocked internally (timers) or externally (counters)
  - Generate interrupts on 0 count
MEMORY INTERFACE

0 1 MEGABYTE ADDRESS SPACE
   16 32K PAGES

0 PAGE REGISTERS SELECT APPROPRIATE PAGE

0 BYTE AND WORD ACCESS

Architecture - 6

SEMICONDUCTOR PRODUCTS DIVISION

MEMORY INTERFACE

CPR   DPR   UPR

Sel  20-bit address

TOP  16-bit data

NEXT

Memory

Architecture - 7

SEMICONDUCTOR PRODUCTS DIVISION
ASIC Bus™

- Data path between Top and other registers and I/O devices
- Input data is fed through ALU before going into Top
- Simple interface for multi-RTX applications

RTX Instruction Set

All processor instructions are 16 bits, with the following general fields:

<table>
<thead>
<tr>
<th>15</th>
<th>12 11</th>
<th>8 7</th>
<th>6 5 4</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>ALU</td>
<td>SC</td>
<td></td>
<td>Data</td>
</tr>
</tbody>
</table>

CLASS - General type of instruction:
- Subroutine call
- Branches and loops
- Math/Logic functions
- Register and I/O access
- Short literal
- Long literals
- Main memory access
- User memory access

ALU - ALU function to be performed.

SC - Subclass. Function depends on Class field.

; - Return bit. When set, causes a Return-from-subroutine

DATA - Depending on Class, indicates shift operation, short literal data, G-space address, or memory address.
RTX Instruction Set

Subroutine Calls

Subroutine call takes place in one clock cycle.
Return-from-subroutine can be part of another instruction.

Branching and Looping

Unconditional branch
Conditional branch based on contents of top register
Conditional branch based on contents of index register with auto decrement of index register.

ALU Operations

+ - swap- and OR NAND NOR XOR XNOR NOT

Shift Operations

16 and 32-bit shifts, either direction
Signed and unsigned shifts can be combined with ALU operations.

Step Math

Divide
Square root

Register-I/O Access

Read/write processor register or I/O device
Combine ALU operation with data from register.

Short Literals

Load value 0 - 31 into top register
Combine ALU operation with data
Literal value is embedded in instruction.

Long Literals

Load 16-bit value into top register
Combine ALU operation with data

Memory Access

Address in top register, data in next register
Access memory by byte or by word
Access program memory, data memory, or user memory
Access memory in either MSB-LSB or LSB-MSB byte order
Combine ALU operation with memory data
Move blocks of memory to/from stack with auto address update

Streamed Instruction Mode

Execute instruction repeatedly without extra fetch cycles

Architecture - 10

SEMICONDUCTOR PRODUCTS DIVISION

Architecture - 11

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000
PROCESSOR PINOUT

MEMORY INTERFACE PINS
ASIC BUS INTERFACE PINS
INTERRUPT/RESET PINS
CLOCK/WAIT PINS

SEMICONDUCTOR PRODUCTS DIVISION

HARRIS RTX2000
INTERFACING THE RTX2000: PROCESSOR PINOUT

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000
MEMORY INTERFACE PINS

MD00 TO MD15: MEMORY DATA BUS
MA01 TO MA19: MEMORY ADDRESS BUS
LDS, UDS: LOWER / UPPER
DATA SELECT (NOT STROBED)
NEW: INSTRUCTION FETCH
BOOT: GENERAL PURPOSE
MR/W: MEMORY READ / WRITE

SEMICONDUCTOR PRODUCTS DIVISION

HARRIS RTX2000
ASIC BUS INTERFACE PINS

GD0 - GD15: G-BUS DATA
GA0 - GA2: G-BUS ADDRESS
GIO\: G-BUS STROBE
GR/W: G-BUS READ/WRITE

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000

INTERRUPT/RESET PINS

NMI: NON-MASKABLE INTERRUPT
E11 - E15: EXTERNAL INTERRUPTS
E13 - E15: TIMER/COUNTER INPUTS
INTSUP: INTERRUPT SUPPRESS
INTA: INTERRUPT ACKNOWLEDGE
RESET: MASTER RESET

SEMICONDUCTOR PRODUCTS DIVISION

HARRIS RTX2000

CLOCK/WAIT PINS

WAIT: WAIT STATE INPUT
ICLK: INPUT CLOCK
TCLK: TIMING CLOCK
FREE RUNNING
PCLK: PROCESSOR CLOCK
HELD LOW FOR WAIT STATES

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000

MINIMAL SYSTEM TIMING
ALLOWS THE USE OF THE SLOWEST MEMORIES
NO DECODE OR BUFFERING IN ADDRESS OR CHIP ENABLE PATHS
AT MAX SPEED: WE\ DELAYED TO ASSURE ADDRESS SETUP TIME
TAA = TCY - TAV - TRDS

HARRIS RTX2000
INTERFACING TO RAMS: MINIMAL SYSTEM

CRITICAL SPEED PATH: USUALLY ADDRESS ACCESS TIME
TAA = TCY - TAV - TRDS

NOTE:
FOR SPEEDS ABOVE 10MHz, WE IS GATED WITH DELAYED PCLK; PCLK IS AT 10MHz AND BELOW, GATING WITH UNDELAYED PCLK IS SUFFICIENT.
HARRIS RTX2000
INTERFACING TO ROMS: MINIMAL SYSTEM

CRITICAL SPEED PATH: USUALLY ADDRESS ACCESS TIME

<table>
<thead>
<tr>
<th>CRITICAL PATH TIME</th>
<th>LOW CORPORATION</th>
<th>HIGH CORPORATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRG = TCV = TAU = TRG</td>
<td>LOWEST</td>
<td>HIGHEST</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>LOWEST</th>
<th>HIGHEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS VALID</td>
<td>13</td>
<td>23</td>
</tr>
<tr>
<td>ADDRESS DATA</td>
<td>24</td>
<td>24</td>
</tr>
</tbody>
</table>

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000

UNBUFFERED SYSTEM TIMING

ALLOWS GREATER DECODING RANGE
LOWER SYSTEM POWER
ONLY ONE DECODER IN CRITICAL PATH

TAA = TCY - TAV - TPD - TRDS

WE REQUIRES GREATER DELAY AT FULL SPEED

SEMICONDUCTOR PRODUCTS DIVISION

HARRIS RTX2000

INTERFACING TO RAMS: UNBUFFERED SYSTEM

CRITICAL SPEED PATH: USUALLY CHIP ENABLE ACCESS TIME

<table>
<thead>
<tr>
<th>Timing Parameter</th>
<th>Low Speed</th>
<th>High Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCY</td>
<td>2ns</td>
<td>4ns</td>
</tr>
<tr>
<td>TAV</td>
<td>2ns</td>
<td>4ns</td>
</tr>
<tr>
<td>TPD</td>
<td>2ns</td>
<td>4ns</td>
</tr>
<tr>
<td>TRDS</td>
<td>2ns</td>
<td>4ns</td>
</tr>
<tr>
<td>TAGE</td>
<td>2ns</td>
<td>4ns</td>
</tr>
</tbody>
</table>

NOTE:
ADDRESS MAY BE BUFFERED ONE LEVEL WITHOUT A SPEED PENALTY.
WRITE ENABLE MAY BE CACHED WITH UNDELAYED POLC WHEN FREQUENCY IS LESS THAN OR EQUAL TO 6MHz.

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000
BUFFERED SYSTEM TIMING

FULLY BUFFERED FOR LARGE MEMORY ARRAYS
DATA BUFFER DELAY IN CRITICAL SPEED PATH
BOTH WE\ AND DATA BUFFER ENABLE DELAYED FOR WRITE CYCLE AT FULL SPEED
TAA = TCY - TAV - TDD - TBD - TRDS

SEMICONDUCTOR PRODUCTS DIVISION

HARRIS RTX2000
INTERFACING TO RAMS: BUFFERED SYSTEM

CIRCUIT DIAGRAM

CRITICAL SPEED PATH: USUALLY CHIP ENABLE ACCESS TIME
TAA = TCY - TAV - TDD - TBD - TRDS

NOTE:
ADDRESS MAY BE BUFFERED ONE LEVEL WITHOUT A SPEED PENALTY.
WRITE ENABLE MAY BE ACTED WITH UNDELAYED PCLK WHEN FREQUENCY IS LESS THAN 8 OR EQUAL TO 1MHz.

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000
INTERFACING TO ROMS: BUFFERED SYSTEM

CHIP ENABLE DECODER

OUTPUT ENABLE DECODER

DATA BUS BUFFER: 1 OF 2

LOW BYTE ROM

HIGH BYTE ROM

CRITICAL SPEED PATH: USUALLY CHIP ENABLE ACCESS TIME

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value 1</th>
<th>Value 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCLK HIGH TIME</td>
<td>10ns</td>
<td>20ns</td>
</tr>
<tr>
<td>POA: POCL HIGH TO ADDRESS VALID</td>
<td>50ns</td>
<td>50ns</td>
</tr>
<tr>
<td>TLB: DECODE DELAY TIME</td>
<td>50ns</td>
<td>50ns</td>
</tr>
<tr>
<td>TBO: BUFFER DELAY TIME</td>
<td>10ns</td>
<td>10ns</td>
</tr>
<tr>
<td>TDK: READ DATA VALID TO POCL HIGH</td>
<td>10ns</td>
<td>10ns</td>
</tr>
<tr>
<td>TDA: MEMORY ADDRESS ACCESS TIME</td>
<td>20ns</td>
<td>20ns</td>
</tr>
</tbody>
</table>

NOTE: ADDRESS MAY BE BUFFERED ONE LEVEL WITHOUT A SPEED PENALTY.

POLY DELAY CIRCUIT

SEMICONDUCTOR PRODUCTS DIVISION

HARRIS RTX2000
BUFFERED SYSTEM: 10MHz TIMING

SEMICONDUCTOR PRODUCTS DIVISION
HARRIS RTX2000
ASIC BUS INTERFACE

OUTPUT PORT
STROBED WITH G10\TIMING NOT CRITICAL

INPUT PORT
TIMING CRITICAL
STROBED EVERY READ CYCLE
ON 74F540 OR 74AC540

--- SEMICONDUCTOR PRODUCTS DIVISION ---

HARRIS RTX2000

TYPICAL ASIC BUS INTERFACES

OUTPUT PORT

INPUT PORT

--- SEMICONDUCTOR PRODUCTS DIVISION ---
The RTXDS Development System

Overview

- The environment in which the Harris RT2000 will be used is real-time embedded process control.

- The requirements for these systems as compared to conventional computer systems are unique:

  - Custom designed hardware and unique I/O devices and interfaces.
  - Interrupt driven, real-time and time critical applications.
  - ROM based code and limited availability of RAM and ROM storage.
Development Systems

- Embedded (or target) systems are generally inappropriate environments for the development of applications software.

- The target system most often lacks support for program development and testing:
  - No auxiliary storage (disk),
  - No interface to a human operator (CRT/keyboard),
  - Limited main storage (RAM),

- Lack of mature software support and development tools.

Testing and Fault Isolation

- Excessive cost, in both dollars and in elapsed time, associated with the implementation of software has become of overriding concern in the design and development of computer systems.

- The software development phase most often underestimated, leading to cost and time overruns, is that of software testing and fault isolation.

- These problems are compounded in the testing of complex real-time embedded systems.
Testing of Embedded Systems

Development and testing of sophisticated embedded systems is often exceptionally costly and time consuming.

Both the hardware as well as the software in such systems may be unique and undergoing development. The system is being integrated as testing is conducted. As a result, faults are difficult to isolate.

The demands of a real-time, interrupt driven systems, often with multi-tasking, may introduce race conditions that can be exceptionally difficult to reproduce and to debug.

The programmer or operator has little visibility into the system under test. Unlike more conventional computer systems, the responses of an embedded system often reflect second or third level effects that may be near useless in isolating programming errors.

There are few peripheral devices on an embedded system that are useful in testing. There is no operator interface (CRT/keyboard) and limited internal storage.

A digital analyzer is a useful tool but is sometimes awkward because of the microscopic level of detail provided.

Lastly, there is no on-board software to facilitate testing of the programs.
A DEVELOPMENT ENVIRONMENT FOR EMBEDDED SYSTEMS

GOALS

The system must provide a FORTH interactive support environment.

The tools should be integrated to form a seamless development environment.

The programmer should be able to use symbolic references.

The target processor must be able to operate at full speed.

THE HOST

0 To build applications software in a productive and cost effective manner, we have chosen to implement much of the software support environment on a host computer system.

0 The approach is an accepted and effective means of implementing embedded computer systems.

0 The host and target are connected through a serial interface.

0 The approach allows a unique software design environment: one that is a reliable combination of hardware and software—and one that is rich in both hardware facilities and in software support tools.
THE HOST

- We selected as a host the IBM (or compatible) PC with a full complement of facilities:
  - Extensive internal storage,
  - High volume external (disk) storage,
  - Convenient human interface (CRT/keyboard),
  - A sophisticated software build environment and mature support tools (LMI's PC/FORTH).

THE TARGET

- The target system can be any board containing the Harris RTX 2000 processor along with minimal support hardware.
RTXDS Structure

- PC/FORTH kernel
- RTXDS module
  - Editor
  - Compiler
  - Disassembler
  - Debugger
  - File Interface

Development Tools

- In the RTXDS development environment, Harris provides a complete set of programming support tools to aid the programmer.

TFORTH

The TFORTH cross compiler provides the software development engineer with a system to conveniently handle source programs and to generate readable object code for the processor.

DISASSEMBLER

The disassembler is a software debug tool that converts binary object code into a sequence of the corresponding FORTH code.

FILE INTERFACE

This set of modules provide the programmer with convenient access to disk files.
HOST/TARGET SERIAL LINK

0 A direct serial link from the host to the target facilities the software build process.

0 The link provides a means of transferring object code from the host to the target to facilitate rapid prototyping of applications software.

0 It also allows us to link the two subsystems in such a way that we not only provide the design engineer with an effective means for building source programs and generating target code, but in addition the link supports a powerful testing and fault isolation capability.

RTXDS

VISIBILITY AND CONTROL

0 A major function of RTXDS is to provide a programmer/operator with visibility into a control of the system under test. The system provides a powerful set of utilities to support the application developer in debugging software.

0 The debug facilities are designed and implemented using the host and target subsystems as an integrated software development facility — taking advantage of the capabilities of each subsystem in the design.
THE TFORTH CROSS COMPILER IS AS CLOSE AS POSSIBLE TO A FORTH-83 SYSTEM.

THE TFORTH EMULATOR PROVIDES A METHOD OF TESTING APPLICATION SOURCE CODE WITH PC/FORTH ON THE HOST PC.

THE FACILITY IS USEFUL FOR PRELIMINARY TESTING OF AN APPLICATION.

THE EMULATOR MODELS THE BEHAVIOR OF THE RTX REGISTERS AND THE ASIC BUS.

THE SYSTEM PROVIDES HIGHLY INTERACTIVE CONTROL TO THE PROGRAMMER TESTING SOFTWARE ON THE TARGET SUBSYSTEM.

THE PROGRAMMER HAS THE CAPABILITY TO TEMPORARILY STOP THE EXECUTION OF PROGRAMS IN A UNIT TEST, AND TO EXAMINE AND CHANGE THE STATUS OF THE TARGET SYSTEM.

PROGRAMMER CONTROL IS AT THE CONSOLE OF THE HOST. THE PROGRAMMER IS ABLE TO SELECTIVELY TRACE THE ACTIONS OF PROGRAMS EXECUTING ON THE TARGET, TO EXAMINE AND TO CHANGE PROCESSOR REGISTERS AND STORAGE ON THE TARGET SUBSYSTEM.

REFERENCES TO THE VARIABLES AND DEFINITIONS WITHIN THE TARGET ARE SYMBOLIC. A DICTIONARY IS MAINTAINED IN THE HOST TO MAP THE SYMBOLS TO THE TARGET ADDRESSES.
HARRIS

CONSOLE COMMANDS

>0 >C0
>! >C!
>.S >MOVE >CMOVE
>DROP >DUMP >EXECUTE
>FILL >. >U.
>..REG >..RR >PUSH >POP
>SET.BREAK >CLR.BREAK
>DOWNLOAD >UPLOAD
>G0 >G! >G0
>LOAD
>

RTXDS - 16
SEMICONDUCTOR PRODUCTS DIVISION

HARRIS

RTXDS

VISIBILITY INTO THE TARGET

- Display facilities through the host allow the programmer to view the progress of execution of a program under test on the target subsystem.

- These features are incorporated into the TFORTH cross-compiler as standard FORTH functions.

- The features are implemented such that these debug utilities generate neither execution overhead nor storage requirements in the final target system.

RTXDS - 17
SEMICONDUCTOR PRODUCTS DIVISION
**FORTH Display Utilities**

- `.` `EMIT`
- `CR` .
- `.S .REG`
- `DUMP`

**Memory Reference Trace**

`>RANGE`

---

**Breakpoint Facility**

- We assume that in a test environment, that code as well as data will reside on the target subsystem in RAM.

- We include a capability to allow the programmer to insert a breakpoint into the code being tested.

- When a breakpoint is encountered during program execution, a transfer to the monitor is initiated automatically that allows the operator to gain control at the host console.

- The operator has the option to:
  - Examine the target system status,
  - Resume execution, or
  - Cancel the breakpoint.
RTXDS

SUMMARY

0 RTXDS PROVIDES THE PROGRAMMER WITH AN INTEGRATED FORTH ENVIRONMENT.

0 IT IS SPECIFICALLY DESIGNED TO SUPPORT THE DEVELOPMENT OF REAL-TIME EMBEDDED SYSTEMS.

0 OUR EXPERIENCE INDICATES THAT RTXDS OFFERS A FLEXIBLE AND PRODUCTIVE ENVIRONMENT FOR THE PROGRAMMER.
HARRIS RTX SEMINAR

REAL TIME EXPRESS DEVELOPMENT BOARD

( RTXDB )

TIM Dwyer
LEAD ENGINEER
HARDWARE SUPPORT DEVELOPMENT

SEMI CONDUCTOR PRODUCTS DIVISION

Overview

0 Features
0 Block Diagram
0 Memory Map
0 I/O Ports
0 Serial Port
0 Prototype Areas
0 Board Operation
FEATURES

- Based on the Harris RTX2000 Processor
- Flexible and expandable
- 16K by 16 zero wait state static RAM
- 8K by 16 system PROM
- Up to 8K by 16 user PROM
- 16-bit input and output ports
- High speed serial port
- Prototype areas
MEMORY MAP

First Four Pages Fully Decoded

Real Time Express Development Board 5 of 11
I/O Ports
- Three 16 bit Output Ports (Latched)
- Three 16 bit Input Ports (Sampled)

Serial Port
- Standard Baudrates up thru 19,200 Baud
- Status LEDs for data and handshake signals
- Header allows connector pinout to be changed
Prototype Areas
- Allows RTXDB to be used as foundation for users prototype
- Allows expansion of RTXDB
- 20% of total board area
- Memory Area
  - Header with memory address, data, and control signals
  - Up to four 28-Pin devices
- ASIC Bus Area
  - Header with address, data and control signals
  - 24 by 28 array of plated holes for pin grid array and other package types
  - Dual inline package (DIP) area for standard logic in DIPs

Board Operation
- Green LED
  - Power on, fuse intact
- Reset Switch
  - RTX2000, 82C50A, parallel output ports
- Red LED
  - Blinking - after reset, RTXDB waiting for HOST to establish communication
  - On - RTXDB waiting for a command from the HOST
  - Off - RTXDB is executing instructions
- RS232 Connector
- Power Connector
TForth Compiler

- Forth-83 Compatible
- Cross-compiles to RTX machine code
- Optimizes for RTX architecture
- Designed to support embedded control systems
  - Headerless code
  - Debugging support
TForth Features

- Directives control compiler environment
- Compiles in separate blocks to support interrupts, vector tables, etc.
- Provisions for generating custom RTX instructions
  \text{EB42 UC\text{ODE 02+} \Rightarrow DUP \@ SWAP 2+}
- Can be used interactively from host monitor

TForth Memory Structure

- Compiles code into PC memory segment
- Maintains headers in separate Target Dictionary
TForth Memory Map

PC Memory

- PC/FORTH dictionary
- TForth
- Editor
- Debugger

- PC/FORTH stack

Target memory

- Target Monitor
- RAM

- Target dictionary

- Target code buffer

Compiler Directives

- **ALL DIRECTIVES HAVE DEFAULTS, SOME CONFIGURABLE**

- **COMPILER INVOCATION**
  - START-TFOR TH - TURNS COMPILER ON
  - END-TFOR TH - TURNS COMPILER OFF

- **MEMORY CONFIGURATION**
  - RAM-ONLY - ALL RAM
  - RAM/ROM - SEPARATE MEMORY SPACES

- **MEMORY ADDRESSING**
  - ROMORG - SET CODE ADDRESS
  - RAMORG - SET DATA ADDRESS
TForth Example

\ TForth Example 1
\ Calculates 2a + b - c
START-TFORTH
HEX 0000 ROMORG
: WORD1 ( a b c -- res )
   - SWAP DUP ++ ;
: WORD2 2 3 4 WORD1 ;
END-TFORTH

DISASSEMBLER

- Decomiles RTX code to Forth primitives
- Useful for instruction sequencing and timing
### DISASSEMBLER INVOCATION

- `0 ADDR COUNT DASM` disassembles one word.
- `0 ADDR DASM` disassembles one word.
- `0 FULL DASM` disassembles all compiled code.

---

### DISASSEMBLER EXAMPLE

<table>
<thead>
<tr>
<th>Address</th>
<th>Opcode</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>: WORD1</td>
<td></td>
</tr>
<tr>
<td>0002</td>
<td>AC40</td>
<td></td>
</tr>
<tr>
<td>0004</td>
<td>A0C0</td>
<td>DUP</td>
</tr>
<tr>
<td>0006</td>
<td>A840</td>
<td>+</td>
</tr>
<tr>
<td>0008</td>
<td>A860</td>
<td>+ ;</td>
</tr>
<tr>
<td>000A</td>
<td>BE42</td>
<td>LIT 02</td>
</tr>
<tr>
<td>000C</td>
<td>BE43</td>
<td>LIT 03</td>
</tr>
<tr>
<td>000E</td>
<td>BB44</td>
<td>LIT 04</td>
</tr>
<tr>
<td>0010</td>
<td>0000</td>
<td>WORD1</td>
</tr>
<tr>
<td>0012</td>
<td>A020</td>
<td>;</td>
</tr>
</tbody>
</table>

Call 0000 '...'
RTX Extensions

- Registers and peripherals
  - IBC0 - reads Interrupt Base/Control Register
  - TC11 - writes value to Timer/Counter 1

-Interrupts
  - Directives for generating vector tables
  - SOFTINT - software interrupt

- ASIC Bus
  - G G0 - reads data from port G
  - N G G! - writes data to port G

Optimization

- Combines subroutine return with last word of definition

- Combines ALU and shift operations with previous instruction
  
- Combines stack operations
  - SWAP DROP = NIP
Optimization Example

\ Optimization example
START-TFORTH
HEX 0000 ROMORG
: WORD3 2 SWAP - SWAP DROP DUP ;
: WORD4 18 G@ + ;
END-TFORTH

0000 : WORD3
     BCC2 LIT 02 SWAP -
0002 : AOAO SWAP DROP DUP ;
0004 : WORD4
     8888 18 G@ + ;

TForth - 12
SEMICONDUCTOR PRODUCTS DIVISION

Debug Support

0 TERMINAL I/O supported through target monitor
   " EMIT KEY

0 CONDITIONAL-compilation
   SWAP {{ ." Executing READ-DATA. Stack: " .S }} 2*

0 MEMORY-access trace
   ALL MEMORY REFERENCES (Q ! C@ C!) are routed through the
   MONITOR
FILE INTERFACE

- WRITE-HEX FILENAME
  Generates Intel Hex file
- WRITE-IMAGE FILENAME
  Generates object file
- READ-IMAGE FILENAME
  Reads image file
- READ-HEX FILENAME
  Reads hex file
- FTYPE FILENAME
  Similar to DOS TYPE command
- FDUMP FILENAME
  Hex/ASCII dump

Exercise

Objective: Write a program which flashes "SOS" in Morse code on the LED on the front of the box.

Hints:
- The LED is turned on by writing 8000H to port 18H: 8000 18 61
- The LED is turned off by writing 0000H to the port.
- SOS is 3 short flashes, followed by 3 long flashes, followed by 3 short flashes
- Gentlemen, start your code at 8300H: 8300 6101 62

If you have time
Write the following so that they may be compiled with either pc/Forth or tforth.

1. Write a program to calculate the Nth Fibonacci number Fn:
   \[ F_0 = 0, F_1 = 1 \]
   \[ F_k = F_{k-1} + F_{k-2} \text{ for } k > 1 \]

2. Write a program to calculate \( N! \) for any \( N, \) \( 0 \leq N \leq 8 \)