IEEE-copyrighted material Proc. FTCS-28, '98
Fastabstracts, 23-25 June, 1998 Munich, Germany

FTSM: A Fault-Tolerant Spaceborne Microcontroller

D. W. Caldwell, D. A. Rennels
Department of Computer Science, 4731 Boelter Hall
University of California, Los Angeles, CA 90024

The search within the aerospace industry to dramatically reduce the cost of high-performance spacecraft makes the use of non-hardened, commercial microcontrollers extremely attractive. These very inexpensive devices are highly integrated computer systems on a chip: a processor and various support functions such as program memory, scratchpad RAM, discrete I/O, A/D converters, serial ports, and counter/ timers. Commercial microcontrollers have not been widely used in space because of their low radiation tolerance -- resulting in a very high rate of transient errors and a potential for circuit latchup [1]. As it is not cost-effective to modify these devices, fault-tolerance is the only option available. FTSM is an architecture, currently implemented in breadboard form and being tested, that is aimed at applying fault-tolerance to deal with these problems and then at developing systems suitable for use in space.

Microcontrollers have little or no built-in fault tolerance features and, since they are very highly integrated, it becomes an interesting design challenge to protect the features already there (e.g., serial buses, programmable bi-directional I/O pins) while using as few on-chip resources as practical in implementing fault-tolerance. A key constraint is to minimize the amount of external support logic so that the main advantage of microcontrollers' high functional density can be maintained.

The block diagram of a spacecraft subsystem (e.g., an IMU) containing an FTSM is shown in Figure 1. Redundant microcontrollers operate in a voted configuration, and they are time-triggered by a common real time interrupt. At the top of the figure, the subsystem interface is shown as just a source of power and communications. The I/O of multiple microcontrollers are combined and protected by I/O isolation and connected with the sensors and actuators. An External Conflict Resolver circuit may reset or power-cycle the individual microcontrollers to support fault recovery.

During normal operation, one microcontroller is the *Master* of the system, while the others provide redundant computation and voting opinions as *Checkers*. The Master and Checkers are loosely synchronized and execute identical application programs, periodically calling support functions which implement the fault-tolerance features. If a microcontroller disagrees with its peers, it can be commanded offline and brought back as a Checker if it can be successfully restarted. Devices are not statically assigned so the operating mode of each device is fluid [2].

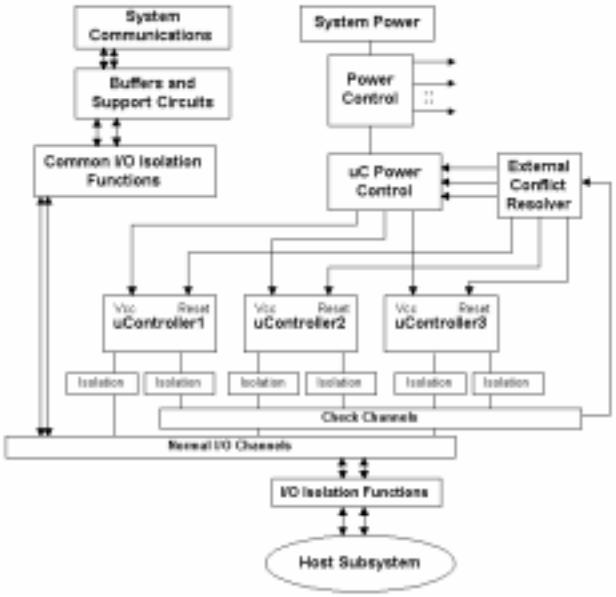

Figure 1: A Subsystem with an Embedded FTSM

Most of the I/O pins of the multiple microcontrollers are bussed together – corresponding pins are connected to a common circuit node through isolation elements (resistors); only the signals governing the external conflict resolution are unique to each processor. This approach simplifies interconnection and makes the architecture easily extensible. Some of the microcontroller's I/O pins are consumed implementing the *Check I/O* necessary for fault-tolerance; the remainder are available to the application as *Normal I/O*. The Check I/O pins provide three functions supporting fault-tolerance: 1) a *Master Channel* (2-pin serial bus) for data communications between microcontrollers, 2) an *Operating Mode Channel* (6-pins) to allow each

microcontroller to broadcast to the others that it is in one of three operating modes, and 3) four *External Resolver control signals* (4-pins) from each microcontroller to request recovery actions. All the check I/O signals use existing I/O functions on the microcontroller and only require external resistive isolation circuits be added.

Each microcontroller, executing identical code, expects check messages to appear on the Master Channel. These messages are then compared locally to detect errors. If it detects an error, it signals the External Conflict Resolver requesting that an action be applied to itself, its left neighbor, its right neighbor, or all the modules. The action request is to either reset or power-cycle the specified processor(s). The External Conflict Resolver carries out an action if a majority of the microcontrollers request that an action be taken against the same device. In order to reduce its susceptibility to transient errors, it is made up of combinational logic.

Circuit isolation is essential in this design: (i) to prevent catastrophic shorts, and (ii) to make it possible to remove power from a module for latchup circumvention. Isolation is provided with resistors in series with the bussed I/O pins of the microcontroller as shown in Figure 2. Pull-down resistors are also employed so that the common output becomes the logical OR of the individual signals.

Power must quickly be turned off when a latchup is detected, but external logic signals can leak past the input protection diodes and still parasitically power the chip enough to fail to clear the latchup condition. In this design when power is removed, Vcc is shorted to ground, and the series resistors divide logic voltages to an acceptably low level.

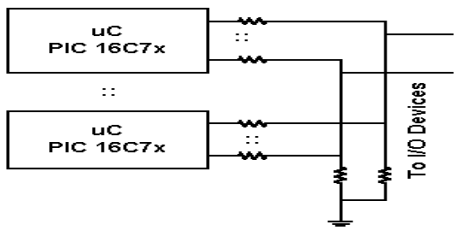

Figure 2: Circuit Isolation

In addition to the conventional ways of protecting outputs (voting outputs at the actuators or strobing data into radiation-resistant latches in the actuators immediately after a successful vote), a novel low-cost masking approach has been developed to protect outputs against single-event upsets. The isolation circuits between microcontrollers produce a logical OR of their individual output signals. If any microcontroller produces a "one" on a bussed output line, the result will be a logic "one". Due to the bi-directional nature of microcontroller pins, two flip flops must be set for an output pin to generate a one--an output 3-state enable and the data latch. Thus we always reset both flip flops for output signals in each microcontroller when a zero is desired. It takes multiple bit-flips in output data and control registers to generate an erroneous output.

An experimental testbed has been constructed based on the 8-bit Microchip PIC 16C7x micro-controller. A small circuit card was constructed with triplicated microcontrollers, a PLD-based external conflict resolver and the resistively isolated interconnections between processors. To simplify initial testing, three in-circuit emulators are used in the circuit board, and the emulators and I/O circuits are connected to a PC that controls and runs fault-insertion experiments on the testbed. The microcontrollers can be interrupted and controlled through their built-in asynchronous serial ports.

A transient fault (simulating a single-event upset) can be inserted in any processor by flipping any bit in any memory location or register and observing the response of the system to the inserted error. Initial test results are very promising. Recovery has occurred in over 98% of tests; erroneous data were prevented from propagating in 99.9% of the tests. In the majority of cases, one processor is voted off-line and computation continues while it is restarted. In approximately 1/3 of the cases the Master is taken off-line requiring a rollback/restart of the system.

We plan to interface the microcontrollers to parts of an inertial measurement unit to provide a realistic application environment in which the hardware and software of the FTSM resides. A future step in this development will be to expose the microcontrollers to a real radiation environment either on the ground or in space to see how well the fault-tolerance techniques work in their planned environment.

*** This work is sponsored at UCLA by the Office of Naval Research under grant #N00014-96-1-0837*

References

1. G. C. Messenger, M. S. Ash. "The Effects of Radiation on Electronic Systems." Second Edition. Van Nostrand Reinhold, New York, 1992.

2. D. W. Caldwell, D. A. Rennels. "A Minimimalist Hardware Architecture for Using Commercial Microcontrollers in Space." 16th Digital Avionics Systems Conference, Irvine, CA. 28-30 Oct 1997.