ITRON Newsletter No.5

ITRON Technical Committee, TRON Association
Katsuta Building 5F, 3-39, Mita 1-chome, Minato-ku, Tokyo 108, JAPAN
TEL: (03) 3454-3191 FAX: (03) 3454-3224

The ITRON Registration System for Products and Applications

The product listed in another page was newly registered in the period from September 1 through November 1, 1993. Details of the product registration system, and an updated list of registered products, can be obtained by contacting the TRON Association. Previously registered products are also listed in ITRON Newsletter Nos.1 to 3.

ITRON-related Publications

Listed in another page are the publications prepared and issued by the ITRON Technical Committee as of October 1, 1993. The ITRON-μITRON Standard Handbook is a one-volume compilation of the μITRON (Ver 2.0) and ITRON2 specifications. Each of the publications can be obtained directly from the sources indicated.

An English-language edition of the μITRON3.0 specification is currently being prepared and will be available on-line when the translation work is completed, along with earlier versions.

Revisions to μITRON3.0 Specification

The μITRON3.0 specification has been revised as indicated in another page. These revisions correct problems that were discovered after the μITRON3.0 Standard Handbook was issued. The version number of the specification with these revisions is Ver 3.01.00.

New Product

The HI8-3H OS recently registered in the ITRON Registration System for Products and Applications is introduced briefly below.

HI8-3H

Product in Outline

Hitachi's HI8-3H real-time OS was developed for use with that firm's H8/300H Series 8-bit single-chip microcontrollers in advanced mode, i.e., supporting 16 MB of memory space[1].

Features

- Conforms to μITRON Specification Ver 2.0.
  - Application programs can readily be ported among different systems running a μITRON-specification OS.
  - Supports a wealth of functions (43 system calls).
- Emphasis on real-time performance
  - Maximum interrupt mask time in the OS: 10.5μs
  - Task switching time (from issuing of `wup_tsk` system call until next task is executed): 35μs
- Highly compact OS
  - An optimum OS configuration can be achieved by selecting only the system calls necessary for the application system.
  - Program size: 0.5KB – 8.9KB
  - The system calls to be used are incorporated in the system simply by linkage with an OS library.
- C language support
  - The entire user program can be written in C.
  - A system call language interface library is provided.
- On-line debugging by in-circuit emulator
  - Task-level debugging is supported, with task control and status reference performed from the ICE. (Hitachi's E7000 and Hitachi Microcomputer System's MY-ICE support this function.)

† This newsletter is reprinted from TRONWARE vol.24 and TRON PROJECT BIMONTHLY No.30.

[1] A version for normal mode, supporting a memory space of 64 KB, is now under development.

Newly Registered Products (Sep. 1, 1993 – Nov. 1, 1993)

Specification	Product Name	Supported Processor	Company
μITRON2.0	HI8-3H	H8/300H Series	Hitachi, Ltd.

ITRON-related Publications

Name	Type	Price	Publisher	ISBN No.
ITRON-μITRON Standard Handbook	Specification(Japanese)	4,800Yen	Personal Media Co.	4-89362-079-7
μITRON3.0 Standard Handbook	Specification(Japanese)	4,000Yen	Personal Media Co.	4-89362-106-8
ITRON/FILE Standard Handbook	Specification(Japanese)	3,000Yen	Personal Media Co.	4-89362-092-4
ITRON Standard Guidebook '92-'93	Textbook (Japanese)	3,500Yen	Personal Media Co.	4-89362-197-6
μITRON Specification Ver 2.01.00.00	Specification (English)	12,000Yen	TRON Association	–
ITRON2 Specification Ver 2.02.00.10	Specification (English)	15,000Yen	TRON Association	–

NOTES:
- Prices do not include consumption tax.
- TRON Association members may purchase Association publications at a special discount price.
- English-language specifications are also distributed free of charge via the Internet. The method for downloading is explained in Newsletter No.2.

Revisions to μITRON3.0 Specification (Ver 3.00.00 → Ver 3.01.00)

1. The following addition is made to the supplementary explanation of system call `cre_mbf` (p.162).

   [Supplementary explanation]

   When some tasks are waiting in a queue to send messages, it is an implementation-dependent matter whether priority is given to the task with the shortest message to be sent or to the task at the head of the queue. Suppose, for example, that task A with a message size of 40 and task B with a message size of 10 are waiting, in that order, to send their message to a message buffer, and that a buffer space of 20 becomes available after another task performs a `rcv_mbf`. In this case it is up to the implementor whether or not to give priority to task B, which has the smaller message size but is not at the head of the queue.

2. The following addition is made to the supplementary explanation of system call `rcv_mbf` (p.173).

   When a message is received from a message buffer to which two or more tasks are waiting to send messages, depending on the size of the message it is possible that two or more of the waiting tasks will be able to send their messages at the same time, exiting from the wait state. In this case the sequence of messages in the message buffer is implementation dependent. Likewise, the sequence of tasks in the ready queue after exiting from the wait state is implementation dependent in the case of tasks having the same priority.

3. The C language interface for the system calls `get_blk`, `pget_blk`, and `tget_blk` is revised as follows (p.239 and p.382).

   ```
   ER ercd = get_blk ( VP *p_blk, ID mplid, INT blksz ) ;
   ER ercd =pget_blk ( VP *p_blk, ID mplid, INT blksz ) ;
   ER ercd =tget_blk ( VP *p_blk, ID mplid, INT blksz, TMO tmout ) ;
   ```

4. The explanation of `E_PAR` error in the error code explanation for system calls `get_blk`, `pget_blk`, and `tget_blk` is revised as follows (p.242).

   `E_PAR` parameter error (`tmout`)≤(-2), `blksz` is negative or illegal)

5. The explanation of `E_PAR` error in the error code explanation for system call `act_cyc` is revised as follows (p.278).

   `E_PAR` parameter error (`cycno` cannot be used, `cycact` is illegal)

6. The explanation of `E_PAR` error in the error code explanation for system call `ref_cyc` is revised as follows (p.280).

   `E_PAR` parameter error (`cycno` cannot be used, the packet address for the return parameter is a value that cannot be used)

7. The following statement is added to the exposition of system call `def_alm` (p.284).

   The definition of an alarm handler is canceled automatically when the designated time occurs and that handler is started.

8. The following addition is made to the supplementary explanation of system call `def_alm` (p.285).

   At the time an alarm handler is started, the definition of that handler is considered as already canceled. Thus if `ref_alm` is used to refer to information on the started handler in the handler, a `E_NOEXS` error will result. It is possible to redefine an alarm handler with the same number in the handler.

9. The explanation of `E_PAR` error in the error code explanation for system call `ref_alm` is revised as follows (p.287).

   `E_PAR` parameter error (`almno` cannot be used or the packet address for the return parameter is a value that cannot be used)

10. The comment on `tmmode` in the `t_dalm` structure of the C language interface is revised as follows (p.394).

    ```
        UINT    tmmode;        /* start time designation mode*/
    ```