Begonnen am	Dienstag, 18. Juli 2023, 16:10
Status	Beendet
Beendet am	Dienstag, 18. Juli 2023, 17:43
Verbrauchte Zeit	1 Stunde 33 Minuten
Punkte	75,63/100,00
Bewertung	**83,20** von 110,00 (**75,63**%)

Frage 1

Richtig

Erreichte Punkte 2,00 von 2,00

Prioritäts Umkehr kann gelöst werden

- [ ] a. durch ein Zeit-Protokoll.
- [ ] b. durch das Zwei-phasen-Protokoll.
- [x] c. durch das Priority Inheritance Protocol. ✓

Die Antwort ist richtig.

Die richtige Antwort ist:
durch das Priority Inheritance Protocol.

Frage 2

Richtig

Erreichte Punkte 2,00 von 2,00

Beim Offline Scheduling

- [x] a. wird Plan wird vor Laufzeit des Systems entwickelt. ✓
- [x] b. erlaubt eine maximale Prozessorauslastung. ✓
- [ ] c. wird der Plan zur Laufzeit entwickelt.
- [ ] d. ist flexibel, ggf. kein optimaler Plan möglich.

Die Antwort ist richtig.

Die richtigen Antworten sind:
wird Plan wird vor Laufzeit des Systems entwickelt.,

erlaubt eine maximale Prozessorauslastung.

Frage 3

Teilweise richtig

Erreichte Punkte 0,67 von 2,00

Welche der folgenden Aussagen bzgl. der Interruptsteuerung trifft zu?

- [x] a. Pegel-gesteuerte Interrupts werden beim Wechsel des Pegels ausgelöst, daher der Name. ✓
- [ ] b. Pegelgesteuerte Interrupts müssen durch Polling des Pegels abgefragt werden.
- [ ] c. Interrupts sind eine Besonderheit von Intel-Microprozessoren. Auf anderen Architekturen kommen POSIX-Signale zum Einsatz.
- [ ] d. Wurde gerade ein Flanken-gesteuerter Interrupt ausgelöst, so muss erst ein Pegelwechsel der Interruptleitung stattfinden, damit erneut ein Interrupt ausgelöst werden kann.

Die Antwort ist teilweise richtig.

Sie haben 1 richtig ausgewählt.
Die richtigen Antworten sind:
Pegel-gesteuerte Interrupts werden beim Wechsel des Pegels ausgelöst, daher der Name.,

Pegelgesteuerte Interrupts müssen durch Polling des Pegels abgefragt werden.,

Wurde gerade ein Flanken-gesteuerter Interrupt ausgelöst, so muss erst ein Pegelwechsel der Interruptleitung stattfinden, damit erneut ein Interrupt ausgelöst werden kann.

Frage 4

Richtig

Erreichte Punkte 2,00 von 2,00

Wie kann man einen nichtunterbrechbaren Kritischen Abschnitt (Non-preemptive critical section) effizient implementieren?

- [x] a. Zu Beginn des Ressourcenzugriffs werden die Interrupts ausgeschaltet und nach Komplettierung desselben wieder erlaubt. ✓
- [ ] b. man speichert den Bereich im NVRAM.
- [ ] c. man legt den Bereich in den Cache.

Die Antwort ist richtig.

Die richtige Antwort ist:
Zu Beginn des Ressourcenzugriffs werden die Interrupts ausgeschaltet und nach Komplettierung desselben wieder erlaubt.

Frage 5

Richtig

Erreichte Punkte 2,00 von 2,00

Beim Design eines Echtzeitsystems muss man berücksichtigen

- [x] a. das Betriebssystem. ✓
- [x] b. die Hardware. ✓
- [x] c. das Kommunikationssystem. ✓

Die Antwort ist richtig.

Die richtigen Antworten sind:
die Hardware.,

das Betriebssystem.,

das Kommunikationssystem.

Frage 6

Falsch

Erreichte Punkte 0,00 von 2,00

Beim RMS Jobs mit kurzer Laufzeit haben eine höhere Priorität ✗

Die Antwort ist falsch.

Die richtige Antwort lautet:
Beim RMS [hängt die Piorität nicht von der Laufzeit eines Jobs ab]

Frage 7

Richtig

Erreichte Punkte 2,00 von 2,00

Was versteht man unter Synchroner Programmierung (Ablaufsteuerung)?

- [x] a. Die einzelnen Aufgaben oder Teilaufgaben werden zu festen Zeiten eingeplant. ✓
- [x] b. Arbeitet nach dem EVA - Prinzip. ✓
- [ ] c. Die Aufgaben werden dynamisch, während der Programmausführung, eingeplant.

Die Antwort ist richtig.

Die richtigen Antworten sind:
Die einzelnen Aufgaben oder Teilaufgaben werden zu festen Zeiten eingeplant.,

Arbeitet nach dem EVA - Prinzip.

Frage 8

Richtig

Erreichte Punkte 2,00 von 2,00

Wozu wird ein Einplanungstest genutzt?

- a. ob eine gegebene Taskmenge unter einem beliebigen Schedulingverfahren planbar ist.
- b. ob ein einzelner Task unter einem gegebenen Schedulingverfahren planbar ist.
- c. ob eine gegebene Taskmenge unter einem gegebenen Schedulingverfahren planbar ist. ✓

Die Antwort ist richtig.

Die richtige Antwort ist:
ob eine gegebene Taskmenge unter einem gegebenen Schedulingverfahren planbar ist.

Frage 9

Richtig

Erreichte Punkte 2,00 von 2,00

Wozu werden Realtime Systeme eingesetzt?

- a. Für interaktive realtime- Benutzer.
- b. Zur Überwachung von Ereignissen. ✓
- c. Vorrangig für Mainframe-Computer genutzt.
- d. Für die Programm Entwicklung.

Die Antwort ist richtig.

Die richtige Antwort ist:
Zur Überwachung von Ereignissen.

Frage 10

Teilweise richtig

Erreichte Punkte 0,67 von 2,00

Welchen Vorteil haben Mutex im Vergleich zu binären Semaphoren?

- [ ] a. Mutexe realisieren einen wechselseitigen Ausschluss/Schutz einer *Critical Section*.
- [ ] b. Mutexe sind einfach zu implementieren.
- [x] c. Mutexe realisieren die Ownership eines Synchronisationsobjekts. ✓

Die Antwort ist teilweise richtig.

Sie haben 1 richtig ausgewählt.
Die richtigen Antworten sind:
Mutexe sind einfach zu implementieren.,

Mutexe realisieren einen wechselseitigen Ausschluss/Schutz einer *Critical Section*.,

Mutexe realisieren die Ownership eines Synchronisationsobjekts.

Frage 11

Vollständig

Erreichte Punkte 15,00 von 15,00

Ist das folgende Taskset (Periodische Tasks, Ti (C, D, P)) ausführbar mit RMS?

T1 (1, 5, 5)
T2 (4, 8, 4)
T3(5, 9, 6)

Die Antwort ist zu begründen. Rechenweg muss nachvollziehbar sein.

U(123) = C(T1)/P(T1)+C(T2)/P(T2)+C(T3)/P(T3)
U(123) = (1/5)+ (4/9)+ (2/6), = 0.977
CPU Auslastung ist 97.7%, es könnte also ausführbar sein.

Einplanungstest für RMS nach Liu/Layland

B(3) = 3 * (2^(1/3) -1) = 0.779
U(123) st größer als B(3)
Präziser Test ist notwendig.

Einplanungstest für RMS nach Liu/Layland für Task 1 und 3
U(13) = (1/5)+(2/6) = 0.533
B(2) = 2 * (2^(1/2) -1) = 0.828
U(13) ist kleiner als B(2), dieses Taskset wäre ausführbar.

Präziser Test für T2 notwendig:

R3(0) = C1 + C2 + C3
  = 1 + 2 + 4 = 7
R3(1) = C3 + aufgerundet(R3(0) / T1) *C1 + aufgerundet(R3(0) / T2)*C2
  = 4 + aufgerundet(7 / 5) + aufgerundet(7 / 6) *2
  = 4 + 2 + 4 = 10
R3(1) ist größer als die Periodendauer von T2, das Set ist nicht ausführbar.

Kommentar:

UB-Test ist hier nicht anwendbar (Folgefehler zu DMS Frage).

Der Precise-Test muss für T3 und T2 angewendet werden.

Reihenfolge der Task stimmt, Precise Test unter der falschen Annahme ok

Frage 12

Richtig

Erreichte Punkte 10,00 von 10,00

In welcher Reihenfolge werden beim folgenden Beispiel die Ausgaben erzeugt?

```
#include <sched.h>
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>

pthread_mutex_t mutex;
pthread_cond_t cond;

void * print_hello(void * arg)
{
  struct timespec now;
  struct timespec timeout;

  printf("<child>: Hello World! task with max priority \n");
  clock_gettime( CLOCK_REALTIME, &now );

  printf("\nnow tv_sec = %d, tv_nsec = %d\n", now.tv_sec, now.tv_nsec);

  timeout.tv_sec  = now.tv_sec + 3;
  timeout.tv_nsec = now.tv_nsec;

  printf("timeout tv_sec = %d, tv_nsec = %d\n", timeout.tv_sec, timeout.tv_nsec);
  printf("The task is coming to enter in a timed wait\n");
  pthread_cond_timedwait(&cond, &mutex, &timeout);
  printf("The task is coming out from the timed wait \n");
  return NULL;
}

void * print_hello_a(void * arg)
{
  printf(" <child>: Hello World! Task with lowest priority ");
  return NULL;
}


int main(int argc, char **argv)
{
  pthread_attr_t      attr;
  pthread_t           child1;
  pthread_t           child2;
  struct sched_param  param;

  (void) argc;
  (void) argv;

  pthread_attr_init(&attr);
  pthread_attr_setschedpolicy(&attr, SCHED_FIFO);
  pthread_mutex_init( &mutex, NULL );
  pthread_cond_init( &cond, NULL );

  printf("<main> Enter in the main \n");

  printf("Creating first task \n");
  param.sched_priority = sched_get_priority_max(SCHED_FIFO);
  pthread_attr_setschedparam(&attr, &param);
  if ( pthread_create( &child1, &attr, print_hello, NULL) ||
       pthread_setschedparam(child1, SCHED_FIFO, &param) ) {
    printf(
      "Thread cannot be created or you have not enough privileges \n"
      "    to set priority!!!!\n");
    exit(1);
  }

  printf("First Task created \n");
  printf("Creating second task \n");
  param.sched_priority = sched_get_priority_max(SCHED_FIFO) - 1;
  pthread_attr_setschedparam(&attr, &param);
  if ( pthread_create( &child2, &attr, print_hello_a, NULL) ||
```

```
      pthread_setschedparam(child2, SCHED_FIFO, &param) ) {
    printf(
      "Thread cannot be created or you have not enough privileges \n"
      "    to set priority!!!!\n");
    exit(1);
  }
  printf("Second task created \n");

  printf("<main> Out of the main\n");
  pthread_join( child1, NULL );
  pthread_join( child2, NULL );

  exit(0);
}

#if defined(__rtems__)
#include <bsp.h>

static void *POSIX_Init()
{
  return (void *)main(0, NULL);
}

#define CONFIGURE_APPLICATION_NEEDS_CONSOLE_DRIVER
#define CONFIGURE_APPLICATION_NEEDS_CLOCK_DRIVER

#define CONFIGURE_MAXIMUM_POSIX_THREADS              10
#define CONFIGURE_POSIX_INIT_THREAD_TABLE

#define CONFIGURE_INIT
#include <rtems/confdefs.h>
#endif
```

a.
...
<main> Out of the main
The task is coming out from the timed wait
<child>: Hello World! Task with lowest priority

b. ✕
...
<main> Out of the main
<child>: Hello World! Task with lowest priority
The task is coming out from the timed wait

...
<main> Out of the main
<child>: Hello World! Task with lowest priority
The task is coming out from the timed wait

c.
...
<child>: Hello World! Task with lowest priority
<main> Out of the main
The task is coming out from the timed wait

Die Antwort ist richtig.

Die richtige Antwort ist:

...
<child>: Hello World! Task with lowest priority

<main> Out of the main
The task is coming out from the timed wait

Kommentar:
Bei Unix-Systemen ok so.

Frage 13

Richtig

Erreichte Punkte 2,00 von 2,00

Welchen Compiler benötigt man wenn Host- und Target- System eine andere Architektur haben?

- [ ] a. Visual-Studio IDE
- [ ] b. Einen C- Compiler
- [x] c. Einen Cross- Compiler ✓

Die Antwort ist richtig.

Die richtige Antwort ist:
Einen Cross- Compiler

Frage 14

Teilweise richtig

Erreichte Punkte 2,80 von 4,00

Damit ein Deadlock entstehen kann, müssen folgende Bedingungen gleichzeitig erfüllt sein

- [ ] a. Rekursive Funktionsaufrufe (recursive calls)
- [x] b. Anforderung weiterer Betriebsmittel (hold and wait) ✓
- [ ] c. Ununterbrechbarkeit (no preemption)
- [ ] d. Geschachtelte *for*-Schleifen (recursive for loops)
- [x] e. Zyklische Wartebedingung (circular wait) ✓
- [x] f. Wechselseitiger Ausschluss (mutual exclusion) ✓
- [x] g. Warteschlangen (waiting queues) ✗
- [ ] h. Häufige Funktionsaufrufe (massive calls)

Die Antwort ist teilweise richtig.

Sie haben 3 richtig ausgewählt.
Die richtigen Antworten sind:
Wechselseitiger Ausschluss (mutual exclusion),

Anforderung weiterer Betriebsmittel (hold and wait),

Ununterbrechbarkeit (no preemption),

Zyklische Wartebedingung (circular wait)

Frage 15

Teilweise richtig

Erreichte Punkte 1,00 von 2,00

Was bedeutet der Begriff *Idle-Task*?

- [x] a. der Task mit der geringsten Priorität. ✓
- [ ] b. sind alle tasks die nicht im *ready* Zustand sind.
- [ ] c. stellt sicher, dass das System noch reagiert ('lebt').

Die Antwort ist teilweise richtig.

Sie haben 1 richtig ausgewählt.
Die richtigen Antworten sind:
der Task mit der geringsten Priorität.,

stellt sicher, dass das System noch reagiert ('lebt').

Frage 16

Vollständig

Erreichte Punkte 10,00 von 15,00

Ist das folgende Taskset (Periodische Tasks, Ti (C, D, P)) ausführbar mit DMS?

T1 (1, 5, 5)
T2 (4, 8, 4)
T3(5, 9, 6)

Die Antwort ist zu begründen. Rechenweg muss nachvollziehbar sein.

Die Deadlines sind kürzer als die Perioden, somit ist, da das RMS mit den selben Perioden und Laufzeiten nicht möglich ist, ist das DMS auch nicht möglich.

U(123) = C(T1)/P(T1)+C(T2)/P(T2)+C(T3)/P(T3)

U(123) = (1/5)+ (4/9)+ (2/6), = 0.977
CPU Auslastung ist 97.7%, es könnte also ausführbar sein.

Einplanungstest für DMS nach Liu/Layland

Udead(132) = (1/5)+ (4/8)+ (2/4) = 1.2
B(3) = 3 * (2^(1/3) -1) = 0.779
Udead(132) ist größer als B(3)
Präziser Test ist notwendig.

Einplanungstest für DMS nach Liu/Layland für Task 1 und 3
Udead(13) = (1/5)+ (2/4) = 0.7
B(2) = 2 * (2^(1/2) -1) = 0.828
Udead(13) ist kleiner als B(2), dieses Taskset wäre ausführbar.

Präziser Test für T2 notwendig: (ist genau der selbe wie bei RMS)

R3(0) = C1 + C2 + C3
  = 1 + 2 + 4 = 7
R3(1) = C3 + aufgerundet(R3(0) / T1) *C1 + aufgerundet(R3(0) / T2)*C2
  = 4 + aufgerundet(7 / 5) + aufgerundet(7 / 6) *2
  = 4 + 2 + 4 = 10
R3(1) ist größer als die Periodendauer von T2, das Set ist nicht ausführbar.

Kommentar:

Bei DMS ist die Prio-Reihenfolge T3, T1, T2

Der UB-Tests ist nur anwendbar wenn D = P. (-5 Punkte)

Frage 17

Richtig

Erreichte Punkte 2,00 von 2,00

Der Einplantest mit der Ratenmonotonen Analyse

- a. hat einen nicht determinstischen Aufwand. ✓
- b. ist exakt. ✓
- c. hat einen hohen Aufwand. ✓

Die Antwort ist richtig.

Die richtigen Antworten sind:
ist exakt.,

hat einen hohen Aufwand.,

hat einen nicht determinstischen Aufwand.

Frage 18

Richtig

Erreichte Punkte 2,00 von 2,00

In [Realtime] ✓ - Betriebssystemen ist die Antwortzeit besonders kritisch.

Die Antwort ist richtig.

Die richtige Antwort lautet:
In [Realtime] - Betriebssystemen ist die Antwortzeit besonders kritisch.

Frage 19

Teilweise richtig

Erreichte Punkte 1,33 von 2,00

Der Einplantest nach Liu/Layland

- [x] a. hat einen niedrigen Aufwand. ✓
- [ ] b. hat einen konstanten Aufwand.
- [x] c. ist pessimistisch. ✓

Die Antwort ist teilweise richtig.

Sie haben 2 richtig ausgewählt.
Die richtigen Antworten sind:
ist pessimistisch.,

hat einen niedrigen Aufwand.,

hat einen konstanten Aufwand.

Frage 20

Richtig

Erreichte Punkte 2,00 von 2,00

Warum werden Mutexe verwendet?

- [x] a. um einen Besitzaspekt zu realisieren. ✓
- [x] b. um Race Conditions zu vermeiden/Ressourcen zu schützen ✓
- [x] c. um wechselseitigen Ausschluss zu erwirken. ✓

Die Antwort ist richtig.

Die richtigen Antworten sind:
um Race Conditions zu vermeiden/Ressourcen zu schützen,

um wechselseitigen Ausschluss zu erwirken.,

um einen Besitzaspekt zu realisieren.

Frage 21

Richtig

Erreichte Punkte 2,00 von 2,00

Harte Realtime - Systeme haben [weniger] ✓ Jitter als Soft-Realtime - Systeme.

Die Antwort ist richtig.

Die richtige Antwort lautet:
Harte Realtime - Systeme haben [weniger] Jitter als Soft-Realtime - Systeme.

Frage 22

Vollständig

Erreichte Punkte 2,00 von 10,00

Schreiben Sie ein C-Programm (threads-example-sync.c) mit dem sie 10 Threads erzeugen die folgende Funktion aufrufen:

```
// ethread: example thread
void *ethread (void *arg)
{
    int my_id = *((int *) arg);
    // Take a nap
    sleep (1);
    // say hello and terminate
    printf ("Thread %d: Hello World!\n", my_id);
    sprintf (ret_status [my_id], "Thread %d: %d", my_id, my_id + 10);
    // pass your id to the thread waiting for you to terminate
    // using pthread_join.
    pthread_exit (ret_status [my_id]);
}
```

Es muss sichergestellt werden, dass die Ausgaben in der Reihenfolge von my_id ausgegeben werden. Es muss im main-task auf die Beendigung der Threads gewartet werden und der von jedem Thread zurückgegebene Status ausgegeben werden.

```
rtems_id   Task_id[ 10 ];         /* array of task ids */
rtems_name Task_name[ 10 ];       /* array of task names */

for(int i =0; i<10;i++){

  Task_name[ i ] = rtems_build_name( 'T', 'A', 'S', i );

  rtems_task_create(
            Task_name[ i ], 2, RTEMS_MINIMUM_STACK_SIZE * 1, RTEMS_DEFAULT_MODES,
            RTEMS_DEFAULT_ATTRIBUTES, &Task_id[ 1 ]
          );

  rtems_task_start( Task_id[i] ,my_id, i )

}

for(int i =0; i<10;i++){

int id pthread_join(Task_id[i]);

printf(%i,id);

}
```

Kommentar:
Die Reihenfolge ist hier nicht gesichert. Es fehlen Synchonisationmechanismen (Semaphores/Mutexes)

Frage 23

Teilweise richtig

Erreichte Punkte 1,50 von 2,00

Race conditions

- [x] a. treten auf, wenn zwei tasks/threads den Wert der gleichen Speicherstelle ändern können. ✓
- [x] b. können durch den konsequenten Einsatz von Sempahoren/Mutexes vermieden werden. ✓
- [x] c. sind schwer auffindbare Programmfehler. ✓
- [ ] d. hängen von dem zeitlichen Verlauf des gesamten Systems ab.
- [ ] e. können auf einem Single-Prozessor-System nicht vermieden werden.

Die Antwort ist teilweise richtig.

Sie haben 3 richtig ausgewählt.
Die richtigen Antworten sind:
treten auf, wenn zwei tasks/threads den Wert der gleichen Speicherstelle ändern können.,

hängen von dem zeitlichen Verlauf des gesamten Systems ab.,

sind schwer auffindbare Programmfehler.,

können durch den konsequenten Einsatz von Sempahoren/Mutexes vermieden werden.

Frage 24

Richtig

Erreichte Punkte 2,00 von 2,00

Was versteht man unter Asynchroner Programmierung (Ablaufsteuerung)?

- [ ] a. Die einzelnen Aufgaben oder Teilaufgaben werden zu festen Zeiten eingeplant.
- [x] b. Die Aufgaben werden dynamisch, während der Programmausführung, eingeplant. ✓
- [ ] c. Arbeitet nach dem EVA - Prinzip.

Die Antwort ist richtig.

Die richtige Antwort ist:
Die Aufgaben werden dynamisch, während der Programmausführung, eingeplant.

Frage 25

Richtig

Erreichte Punkte 2,00 von 2,00

Beim Online Scheduling

- [x] a. wir der Ablauf zur Laufzeit entwickelt. ✓
- [ ] b. wird der Plan wird vor Laufzeit des Systems entwickelt.
- [ ] c. ermöglicht eine maximale Prozessorauslastung.
- [x] d. ist flexibel, ggf. kein optimaler Plan möglich. ✓

Die Antwort ist richtig.

Die richtigen Antworten sind:
wir der Ablauf zur Laufzeit entwickelt.,

ist flexibel, ggf. kein optimaler Plan möglich.

Frage 26

Richtig

Erreichte Punkte 2,00 von 2,00

Die Interrupt Latenzzeit (Latency) bei einem Realtime-System

- [ ] a. muss 0 sein.
- [ ] b. muss maximal sein.
- [x] c. muss minimal sein. ✓

Die Antwort ist richtig.

Die richtige Antwort ist:
muss minimal sein.

Frage 27

Teilweise richtig

Erreichte Punkte 0,67 von 2,00

Eine Memory-Management Unit

- [x] a. erhöht die Zeit um eine Operation zu beenden. ✓
- [ ] b. erhöht die Leistungsaufnahme des Systems.
- [ ] c. erhöht die Kosten des Systems.

Die Antwort ist teilweise richtig.

Sie haben 1 richtig ausgewählt.
Die richtigen Antworten sind:
erhöht die Zeit um eine Operation zu beenden.,

erhöht die Leistungsaufnahme des Systems.,

erhöht die Kosten des Systems.

Frage 28

Falsch

Erreichte Punkte 0,00 von 2,00

Welche Kernel werden bei Realtime - Systemen eingesetzt?

- [ ] a. Non-preemtive kernel.
- [ ] b. Weder preemtive noch non-preemtive kernel.
- [ ] c. Preemtive kernel.
- [x] d. Preemtive oder non-preemtive kernel. ✗

Die Antwort ist falsch.

Die richtige Antwort ist:
Preemtive kernel.

◄ Kursraum online VE

Direkt zu:

Mündliche Prüfung 29. September ►