Ann. For. Res. 65(1): 6584-6596, 2022
ISSN: 18448135, 20652445

ANNALS OF FOREST RESEARCH
https://www.e-afr.org/

RESEARCH ON RESOURCE SCHEDULING MODEL OF CONSTANT TEMPERATURE CONTROL FOR SINGLE CHIP EMBEDDED SYSTEM

Fengqin Ren
Taiyuan University, Taiyuan, China
E-mail: eveline@126.com

Abstract: The traditional control resource scheduling model uses the partitioning of the thermostatic control process to partition the scheduling, which has problems such as low efficiency, poor thermostatic control effect and wasted scheduling resources. In order to solve the above problems, the research constructs a thermostatic control resource scheduling model for microcontroller embedded systems. Using a combination of hardware counters and task heat feature portrayal criteria, we obtain the temperature data of the microcontroller embedded system and analyze the thermostatic control load characteristics of the microcontroller based on the data. After predicting the resource scheduling capacity using the parallel decision tree algorithm, the thermostatic control resource scheduling model is constructed. The comparison test data with the two traditional models show that the average maximum temperature of the microcontroller applying this scheduling model is 60.3°, which is lower than the two traditional models, and has the advantages of short duration of maximum temperature and low energy consumption of the microcontroller.

Keywords: microcontroller; embedded system; thermostatic control; resource scheduling; scheduling model;

0 Introduction

Microcontroller embedded system that is embedded microcontroller, refers to the microcontroller as the core control unit embedded in the object system of a dedicated computer system, is a very widely used embedded system structure. Embedded system is the core of the application, based on computer technology, software and hardware can be cut, adapt to the application object environment on the function, reliability, security, cost, volume, weight, power consumption, environmental and other aspects of strict requirements for processing system. As the core of the microcontroller embedded system, the microcontroller from the architecture to the instruction system is specially designed in accordance with the application characteristics of embedded systems, which can best meet the requirements of the embedded, reliable operation and excellent control functions in the field in the face of control objects, application systems [1]. Since microcontrollers have a dedicated architecture and instruction system for embedded system applications, systems and products that can meet the requirements of a variety of different application systems can be derived from their basic architecture. Users can choose the best model of microcontroller according to various different requirements and functions of the application system. Only by a single microcontroller chip is not a single chip embedded system. The core control chip of the system often needs to be organically connected with some peripheral chips, devices and control circuits to form an actual microcontroller system, which is then embedded in

the environmental system of the application object as the core intelligent control unit and constitutes a typical single-chip embedded application system [2-3]. With the development of related technologies, advances in microprocessor design technology have led to a greater density of transistors on the chip and higher performance, and the power consumption of the processor has also increased with the increase in transistor density, but the increasing power consumption and the accumulation of heat on the chip have also caused increasingly serious heat dissipation problems. The high temperature on a microcontroller chip not only reduces the availability of the processor and shortens its life cycle, but also increases the cost of cooling the processor. Processor power consumption problems with Moore's Law shows exponential growth, how can provide a good cooling technology has become an important aspect of computer system design. Some cooling technology, like heatsinks, fans, etc. is still the basic method of heat removal. But these cooling techniques have the disadvantages of high cost and poor practicality [4].

In recent years with the advent of single-chip multiprocessors, multi-threaded processors, and symmetric multiprocessors, the power consumption problem has become more severe due to the greater density of transistors on a single chip. Currently most of the hardware architecture level based heat control techniques are used to enable single-chip embedded systems to maintain the system temperature stability under different operating and processing conditions. This heat control technique monitors the processor temperature by adding a temperature control unit to the processor and slows down the processor to cool down the processor when the processor temperature exceeds the threshold, but also reduces the processor performance [5]. Also, temperature control methods based on the operating system level are increasingly available and are more flexible than architecture-level temperature control, but they also impose higher requirements on control resource scheduling [6-7]. The thermostatic control resource scheduling for microcontroller embedded systems can effectively alleviate the performance degradation of microcontroller embedded systems caused by temperature control techniques for microcontroller embedded systems. The reference [8] uses a migration strategy to balance the heat among multiple processor cores to avoid excessive temperature in one processor core. The model calculates the corresponding temperature hotspots by querying the operation of the internal units of the processor and then determines whether migration needs to be implemented. The model requires a large amount of computational load space and cannot be used in highly threaded systems, which has limitations. The reference [9] uses the TAS coldest core algorithm to implement control resource scheduling to schedule processes to run on the coldest core, and the TAS algorithm is able to significantly reduce the performance loss due to high temperatures. However, its drawback is that the model is computationally intensive and requires high hardware requirements for the microcontroller embedded system. Regarding the above analysis, this paper will study the construction of a thermostatic control resource scheduling model for microcontroller embedded systems.

1 Research on thermostat control resource scheduling model for microcontroller embedded system

1.1 Single-chip embedded system temperature data acquisition

The research-oriented thermostatic control resource scheduling process for microcontroller embedded systems first requires access to microcontroller embedded system temperature data. At present, dynamically obtaining the core temperature of the microcontroller processor, there are generally two perspectives: hardware perspective uses hardware-based performance counters or temperature sensors to obtain the microprocessor temperature; software perspective uses parameters such as the access rate of the microcontroller embedded system to perform tasks in relation to temperature changes to obtain the microprocessor temperature [10-12]. Both microcontroller temperature acquisition methods have advantages and disadvantages, and this study combines the two to improve the accuracy of temperature data acquisition in microcontroller embedded systems .

(1) Performance counter

Most current microprocessors support built-in performance technology registers, and different processors generally have different numbers of counters. Some of the commonly used performance counters are as Table 1.

Table 1 Commonly used performance counter indicators

Serial number	Name	Function
1	Cycles	Clock period
2	Non Halted Cycles	Clock cycles in which the processor is not in the stop state
3	Processor Queue Length	Handle the number of threads in the queue
4	Load Instructions	Number of valid Load instructions
5	Store Instructions	Number of valid Store instructions
6	L2 Cache Read Misses	Number of secondary cache read misses
7	Total Cache Misses	Number of misses for internal Cache
8	Float Instructions	Number of floating-point instructions

The hardware events recorded by the processor hardware counter correspond to the relevant hardware components of the processor, for example, the number of instructions decoded can reflect the working condition of the processor decoding components, the number of L1 cache reads can reflect the number of L1 cache operations, and the number of instructions submitted can reflect the throughput rate of the whole pipeline. The power consumption of each hardware part of the processor can be regarded as a constant, and the power consumption of each part of the processor can be multiplied by the utilization rate of the part to obtain the power consumption of the processor part in a certain time interval [13]. The power consumption of each component can be linearly added up to obtain the power consumption characteristics of the processor.

The following is the way to calculate the power consumption by performance calculator.

The power consumption of a single execution core of a microcontroller embedded system can be expressed as.

$$E_i = \sum_{j=1}^{n} a_j c_j \quad (1)$$

In formula (1), a_j is the power consumption generated by event j; c_j is the value of the performance counter corresponding to event j, and i refers to the serial number of the processor of the microcontroller embedded system. Usually, a_j is obtained by repeatedly executing a certain event, yielding a large number of statistics, and then calculating it. The calculation formula is as follows.

$$a = \frac{E_i}{c} \quad (2)$$

In formula (2), E_i represents the processor core power consumption when the program in question is running. c is the value of the performance counter corresponding to the trigger. According to formula (1) and formula (2) the power consumption of the microcontroller process in a certain time period can be calculated, and by means of the exponentially weighted average method based on this approach the power consumption in the next time period can be predicted. The calculation formula is as follows.

$$Contr_i = pE_i + (1-p)Contr_{i-1} \quad (3)$$

In formula (3), $Contr_i$ is the current microcontroller operating process power contribution value; E_i is the calculated power consumption in the current cycle; $Contr_{i-1}$ is the process power contribution value of the previous sampling cycle; p is the weight. This method is able to calculate the current temperature of the processor based on the value corresponding to the counter, however, it is based on a mathematical model rather than an actual system, and thus is subject to a certain degree of deviation [14].

(2) Task heat feature engraving criteria to calculate microcontroller temperature

The proportion of under-chip instructions mainly reflects the proportion of under-chip instructions among all instructions executed by a task [15]. If the proportion of off-chip instructions among all instructions executed by a task is not high, it means that the processor utilization of this task is not high, and the processor will be mostly idle, which means that the heat generated by the processor will be very low. The proportion of off-chip instructions is calculated as follows.

$$OCIP = \frac{I_{load} + I_{store} + I_{trans}}{I_{total}} \quad (4)$$

In formula (4), I_{total} is the total number of instructions. I_{load} is the all load memory instructions that involve off-chip execution. I_{store} is the number of memory instructions written. I_{trans} is the number of data transfer instructions.

The number of processor cycles per instruction mainly reflects the number of cycles consumed by the processor to complete each instruction [16]. The reason why the standard can reflect the heat characteristics of the task is that different instructions of the processor require different cycles to run, such as add instructions, subtract instructions and shift instructions, which can be completed only inside the processor and require very few cycles, usually only a few cycles, but for instructions involving Memory operation instructions, or inter-chip data transfer instructions, these instructions are required to cooperate with external devices to complete, and the high speed of the processor and the low speed of external devices between a large gap, so it caused the processor had to stop waiting for external devices to complete the operation, so if the CPI is higher, the busier the processor operation, the more heat generated by the processor. The number of processor cycles per instruction is calculated as follows.

$$CPI = \frac{C_{total}}{I_{total}} \quad (5)$$

In formula (5), I_{total} is the total number of completed instructions during the task run. C_{total} is the total number of cycles the processor ran during the task run. Both of these parameters can be obtained by counting the clock cycles of the processor and the number of completed instructions in the online statistical system.

The off-chip access rate reflects the ratio of the number of real memory accesses to the total number of data requests when the task needs to access data in memory, and reflects the task's thermal characteristics. If most of the data fetched by the processor can be cache hit, then the processor can move on to the next instruction as soon as possible without having to spend a long time waiting for the memory to complete the data operation, the processor's utilization is also high and the processor's operation speed is greatly improved [17]. It can be seen that cache hits cause the processor to generate more heat.

The off-chip access rate reflects the extent to which a program accesses memory at runtime by fine-grained, and the off-chip access rate is computed by formula (6).

$$OCR = \frac{C_{miss} + T_{miss}}{C_{ref} + T_{ref}} \quad (6)$$

In formula (6), C_{ref} that the total number of cache accesses; T_{ref} that the number of TLB cache accesses; C_{miss} that the total number of cache access misses; T_{miss} that the total number of TLB cache access misses. The off-chip access rate is inversely proportional to the microcontroller temperature. For different models of microcontrollers the hair-trigger proportional function parameters between their running temperature and the off-chip access rate need to be determined by testing. The above two methods are used to obtain the microcontroller embedded system temperature data during operation and analyze the microcontroller load characteristics when the microcontroller embedded system is performing thermostatic control.

1.2 Single-chip thermostat control load characterization

When a microcontroller embedded system is running, each process is assigned a priority level and the controller always selects the process with higher priority [18]. The priority queuing process of thermostatic controller is as Figure 1.

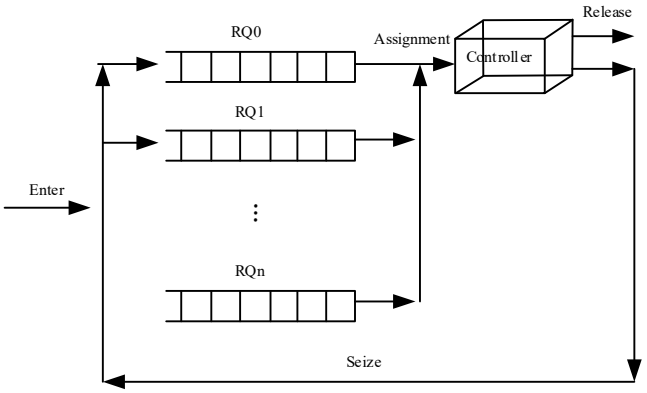

Figure 1 Constant temperature controller priority queuing process

The structure provides a set of queues in decreasing priority order: $RQ_0, RQ_1, RQ_2, ..., RQ_n$, where for all $i < j$, there is priority $[RQ_i] > [RQ_j]$. When making a scheduling selection, the scheduler starts with the queue with the highest priority (RQ_0). If there is one or more processes in this queue, one of them is selected using some scheduling policy; if RQ_0 is empty, RQ_1 is checked and the next processing is similar. At subsequent times, whenever it is preempted, it is demoted to the next lower priority queue. A short process will soon finish executing and a long process will descend step by step. Thus, new arrivals and short processes will take precedence over older and longer processes. A simple FCFS mechanism is used in each queue, except in the lowest priority queue. Once a process is in the lowest priority queue, it will not be lowered again, but will return to that queue repeatedly until the end of the run. Thus, the queue is treated in a rotating manner. A multi-level feedback queue indicates that the operating system assigns a processor to a process, and when that process blocks or is preempted, it is fed back to one of multiple priority queues. To prevent processes in the lower priority queues from starving to death, processes in the lower priority queues are given longer time slices. A multi-level feedback queue indicates that the operating system assigns a processor to a process and when that process blocks or is preempted, it is fed back to one of the multiple priority queues. To prevent starvation of processes in the lower priority queues, processes in the lower priority queues are given longer time slices [19-20]. A typical approach is to allow processes scheduled from RQ_0 to execute one time unit and then be preempted, processes scheduled from RQ_1 to execute two time units, and so on. In general, processes scheduled from RQ_i are allowed to execute for i th power of 2 before they are preempted. Even if a longer time is allocated to a lower priority, it is still possible for a long process to starve to death. It is also possible to promote a process to a higher priority queue when it has been waiting for service in the current queue for more than a certain amount of time. Based on the priority of

the microcontroller embedded system in the thermostat control can determine the load state when the thermostat control, by the different microcontroller load state can determine the thermostat controller load operating characteristics in each state.

In order to describe the operating characteristics of the thermostatic control load, two state variables are introduced: the internal temperature of the microcontroller and the switching state of the load [21]. The thermodynamic operating characteristics of the m th temperature-controlled load during low-power operation of the microcontroller can be expressed as

$$T_m(k+1)=\frac{1}{C_m R_m}\left(T_\infty - T_m(k) - s_m(k) R_m P_m\right) \quad (7)$$

$$s_m(k+1)=\begin{cases} 0, s_m(k)=1 \,\&\, T_m(k) \le T_m^{\min} \\ 1, s_m(k)=0 \,\&\, T_m(k) \ge T_m^{\max} \\ s_m(k), else \end{cases} \quad (8)$$

In formula (7) and (8), T_∞ indicates the external ambient temperature; T_m indicates the indoor temperature, R_m, C_m, P_m indicates the thermal resistance, heat capacity and energy transfer rate of the m th thermostatically controlled load. Considering the dispersion among the temperature control loads, it is assumed that R_m, C_m, P_m obey Gaussian distribution with a standard deviation of 0.1. s_m indicates the working state of the load, and a value of 1 indicates that the temperature control is on, and a value of 0 indicates that the temperature control is off. $T_m^{\max}$ and $T_m^{\min}$ indicate the upper and lower limits of the temperature under high or low load operation, respectively, when the thermostat is controlled by the microcontroller. From the above formula, we can get the thermostat control load characteristic change of the microcontroller embedded system in different operating states. Based on the analysis of the variation of the thermostatic control load characteristics of the microcontroller embedded system, the schedulable capacity of the microcontroller thermostatic control is predicted.

1.3 Thermostatically controlled dispatchable resource capacity forecast

Based on the above analyzed changes in the thermostatic control load characteristics of the microcontroller embedded system, the schedulable resources of the microcontroller embedded system when performing thermostatic control are predicted. To predict the schedulable capacity of the thermostatic control resources of the microcontroller embedded system, it is necessary to mine different characteristics and add different features to the value of this schedulable capacity. The variation of thermostatic control load characteristics of the microcontroller embedded system analyzed in 1.2 and the sequence of resource requirements when the control threads advance under different operating load states of the microcontroller are used as features for the prediction of the thermostatic control schedulable resource capacity.

Based on the selected thermostatic control schedulable resource capacity prediction features, the parallel decision tree algorithm is used to predict the resource scheduling capacity of the microcontroller embedded system. The decision tree easily interprets and handles discrete features,

can be extended to multi-level classification problems without requiring feature data normalization, and can capture the interactions between linear and nonlinear features. The decision tree continuously partitions the feature interval into two parts [22]. By finding the information gain of all the ways that can be partitioned, the feature with the greatest information gain is finally selected as the basis for node partitioning. The optimal feature is selected based on the node impurity. The impurity of a node measures the uniformity of the predictor variables of this node. The calculation of node impurity, including the entropy used to deal with the classification problem, is calculated as follows.

$$S(X)=\sum_{i=1}^{M}-f_{y_i}\lg\left(f_{y_i}\right) \quad (9)$$

In formula (9), f_{y_i} is the frequency of occurrence of the corresponding predictor variable y_i at a node of the decision tree; M is the total number of instances; X is the input feature vector. The information gain is the sum of the impurity of the parent node and the weights of the impurity of its two children [23]. Assuming that the dataset D containing M data is partitioned into two datasets D_{left} and D_{right}, which contain N_{left} and N_{right} data, respectively, then the information gain is calculated as follows.

$$IG(D,s)=S(D)-\frac{N_{left}}{N}S\left(D_{left}\right)-\frac{N_{right}}{N}S\left(D_{right}\right) \quad (10)$$

For the classification problem, the individual decision trees are voted on and the output of the parallel decision tree algorithm is the majority of all decision tree results. The trained parallel decision tree algorithm outputs the adjustable resource capacity predicted for that microcontroller at the time of thermostatic control based on the performance parameters of different types of microcontrollers. The resource scheduling model is constructed based on the control resources predicted by the parallel decision tree algorithm for the microcontroller embedded system at the moment of thermostatic control to be scheduled.

1.4 Complete resource scheduling model construction

For the microcontroller embedded system, the diagram of the microcontroller process conversion process during the operation processing is as Figure 2.

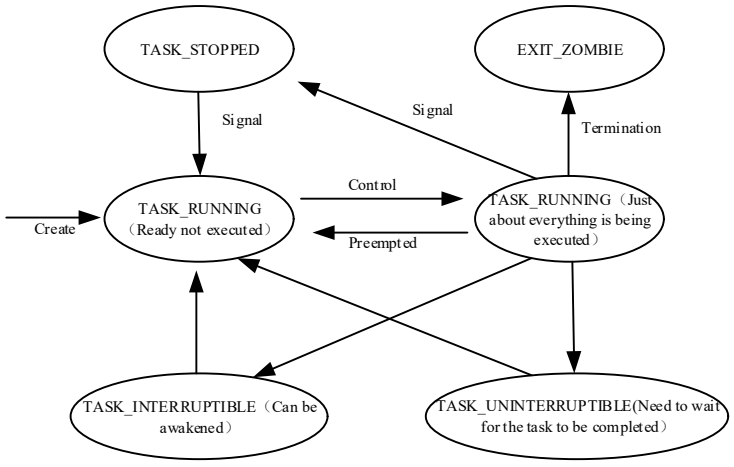

Figure 2 Schematic diagram of microcontroller process state transition

In this paper, we use the temperature-aware principle to construct a thermostatic control resource scheduling model, which will combine the on-chip temperature with the temperature characteristics of the running task, and make full use of the large differences in temperature characteristics between different tasks to develop a thermostatic control resource scheduling policy.

According to the three different microcontroller task heat characteristics portrayal criteria identified in 1.1, OCIP reflects an instruction-level characteristic that determines at a coarse-grained level whether a task is overall a high-temperature task. CPI is initially used to reflect system performance, with a high CPI implying a lighter-running task, which means that the CPU is running at a low speed and a low temperature state reflected from another perspective, CPI reflects the thermal characteristics of a task in more detail than OCIP.

There are four basic principles for constructing a thermostatic control resource scheduling model as follows.

1) When the on-chip temperature is relatively low, try to pick tasks with high temperature (i.e., generating more heat) to run,.

2)When the temperature on the slice is relatively high, try to pick the tasks with low temperature (i.e., generating less heat) to run.

3) When the on-chip temperature exceeds the set temperature gap, run the idle task to cool down the processor.

4) Try to keep the previous scheduling method of task scheduler, such as the priority of tasks, etc.

Then the objective function of the thermostatically controlled resource scheduling model is as follows.

$$P_t\left(p,\varepsilon\right)=P\left(p_h,\frac{T_c-T_{amb}}{T_t-T_{amb}}\right) \quad (11)$$

In formula (11), T_c represents the current on-chip temperature of the microcontroller; T_{amb} represents the current ambient temperature of the microcontroller; T_t represents the current temperature gap value of the microcontroller, and p_h represents the task with the highest priority among the current tasks. $\frac{T_c-T_{amb}}{T_t-T_{amb}}$ indicates the current processor temperature status, if it is greater than 1, it means that the current processor temperature has exceeded the temperature gap value and the processor needs to be cooled down immediately, if it is less than 1, it means that the processor has not exceeded the temperature gap value and is still in a relatively safe temperature region. A higher value of $\frac{T_c-T_{amb}}{T_t-T_{amb}}$ indicates a higher processor temperature and a lower value indicates a lower processor temperature. ε represents the combination of three temperature characteristics, OCIP, CPI and OCR, into one that can overall respond to the resource scheduling

status. The thermostatic control resource scheduling scheme that satisfies formula (11) is the best scheduling. When scheduling, the highest priority value ε and the resource closest to the $\frac{T_c - T_{amb}}{T_t - T_{amb}}$ value are selected for scheduling. This completes the study on the construction of a thermostatic control resource scheduling model for microcontroller embedded systems.

2 Resource Scheduling Model Testing

Keeping the microcontroller temperature constant is an important way to reduce the rapid decrease in the life of the microcontroller. In controlling the stability and constancy of the microcontroller embedded system is mainly by scheduling the control resources inside the microcontroller. Regarding the problems of the traditional control resource scheduling model, the above study constructs a thermostatic control resource scheduling model for microcontroller embedded systems. In this section, the performance test of the control resource scheduling model is carried out to study the feasibility of the model.

2.1 Test data preparation

The multi-core microcontroller is selected as the research object of this test, and the simulation platform receives three input files: .corifig processor parameter configuration file, .flp processor distribution plan, and .ptrace processor power consumption data. Among them, the microprocessor configuration files are all the factory theoretical setting values of this multi-core microcontroller.

2.2 Test content

Multiple control resource scheduling models are selected in the form of comparisons between different scheduling models under the condition of maintaining a constant microcontroller temperature. The resource scheduling model studied in this paper is used as the experimental group, the scheduling model using migration strategy in the reference [8] is the comparison group 1, and the scheduling model based on the TAS algorithm in the reference [9] is the comparison group 2. The comparison angle of the comparison experiment is the energy consumption caused to the chip when the control resource scheduling model is executed and the three indicators of the maximum temperature duration of the microcontroller under different models.

2.3 Test environment

Hardware environment of the test platform.

The data processing platform for this test uses Dell OptiPlex 740 as the data processing host with Intel Core 5 Solo processor, and the specific parameters of this machine are as follows.

1)Intel Core Solo 8560 processor, single-core, 4.25GH, 5890KB L2 cache, bus frequency 1035MHz.

2)Integrated NVIDIA Quadro NVS 5658 chipset.

3)RAM 512M NECC dual-channel DDR2 667MHz SDRAM.

4) Hard disk S ATA (7200RPM - 8MB cache).

2.4 Test results

The comparison of energy consumption and maximum temperature duration of the multicore microcontroller under the scheduling process of three sets of resource scheduling models is as Table 2.

Table 2 Test data

Microcontroller task number	Experimental group model			Comparison group model 1			Comparison group model 2		
	Energy consumption /mW	Maximum microcontroller core temperature /°	Duration /s	Energy consumption /mW	Maximum microcontroller core temperature /°	Duration /s	Energy consumption /mW	Maximum microcontroller core temperature /°	Duration /s
1	3.32	59.3	52.4	7.81	65.3	76.8	12.36	67.6	102.7
2	3.01	60.5	51.1	7.78	64.5	80.5	12.19	67.8	107.4
3	3.31	61.2	54.3	7.60	65.2	73.2	11.23	68.1	94.6
4	2.73	61.3	51.5	7.21	65.6	74.7	11.12	67.7	110.1
5	2.78	60.2	54.7	7.38	64.9	75.6	12.15	67.5	111.2
6	3.14	59.5	54.2	7.34	65.4	77.4	12.75	68.4	111.9
7	2.49	60.2	51.8	7.46	65.1	73.0	11.87	68.9	108.5
8	2.45	60.3	54.9	7.87	64.8	75.3	11.04	67.2	111.6

As can be seen from Table 2, the core temperature of the microcontroller with the experimental group model is lower than the core temperature of the microcontroller with the two comparison group models. The average maximum temperature duration of the microcontroller with the experimental group model is 52.88s shorter than that of the microcontroller with the other two comparison models (the average maximum temperature duration of the comparison group model 1 is 75.63s and the average maximum temperature duration of the comparison group model 2 is 107.23s). The average maximum temperature of the microcontroller applying the experimental group model was 60.3°, the average maximum temperature of the microcontroller applying the comparison group model 1 was 65.2°, and the average maximum temperature of the microcontroller applying the comparison group model was 67.7°. Excluding the influence of factors such as the residual rate of the microcontroller, it shows that the experimental group model has a faster response rate and shorter resource scheduling processing time for the thermostatic control resource scheduling. The energy consumption of the corresponding microcontroller when the experimental group model performs resource scheduling is significantly less than the energy consumption generated by the microcontroller when the other two comparison models perform resource scheduling. In summary, the thermostatic control resource scheduling model for microcontroller embedded system constructed in this paper has the advantages of good response rate, processing efficiency and low energy consumption.

3 Conclusion

This paper constructs a thermostatic control resource scheduling model for microcontroller embedded systems to address the problems of the traditional microcontroller thermostatic control resource scheduling model. The superior performance of this model such as fast processing speed and low energy consumption is verified through comparison experiments with the traditional scheduling model. Since in this study, only the thermostatic control resource scheduling problem in a homogeneous multi-core microprocessor environment is considered, the scheduling model in a heterogeneous multi-core processor environment can be studied in the future to improve the applicability of the thermostatic control resource scheduling model.

Reference

[1] Megantoro P , Anggara F , Prabowo I E , et al. MPPT Technique Based-on Microcontroller Using Combination of Perturb Observe and Constant Voltage Algorithm[J]. Journal of Advanced Research in Dynamical and Control Systems, 2019, 11(8):3268-3277.

[2] Isah M A, Kim B S. Integrating Schedule Risk Analysis with Multi-Skilled Resource Scheduling to Improve Resource-Constrained Project Scheduling Problems[J]. Applied Sciences, 2021, 11(2):650.

[3] Iskrenovia P S, Sretenovi G B, Krsti I B, et al. Thermostat with Peltier element and microcontroller as a driver[J]. Measurement, 2019, 137:470-476.

[4] Cunha M, Viegas J, Martins M, et al. Dual-resource Constrained Scheduling for Quality Control Laboratories, 2019, 52(13):1421-1426.

[5] Jia, Hongjie, Zening, et al. Scheduling distributed energy resources and smart buildings of a microgrid via multi-time scale and model predictive control method[J]. IET Renewable Power Generation, 2019, 13(6): 816-833.

[6] Rezakhah M , Moreno E , Newman A . Practical Performance of an Open Pit Mine Scheduling Model Considering Blending and Stockpiling[J]. Computers & Operations Research, 2019, 115(3):104638.

[7] Vahdani, S. Multi-mode capital-constrained project payment scheduling model considering bonus-penalty structure[J]. International Journal of Management Science and Engineering Management, 2020, 15(1):17-25.

[8] Wrbelauer M , Meyr H , Almada-Lobo B . Simultaneous lotsizing and scheduling considering secondary resources: a general model, literature review and classification[J]. OR Spectrum, 2019, 41(3):1-43.

[9] Wmab C, Shab C, Dmab C, et al. Scheduling decision model of liner shipping considering emission control areas regulations[J]. Applied Ocean Research, 2020,106:102416

[10] Ia A , Gdp A , Gr B . New lower bounds for solving a scheduling problem with resource collaboration[J]. Computers & Industrial Engineering, 2019, 127:225-239.

[11] Noliya A , Kumar S . Performance Analysis of Resource Scheduling Techniques in Homogeneous and Heterogeneous Small Cell LTE-A Networks[J]. Wireless Personal Communications, 2020, 112(4):2393-2422.

[12] Len A , Fernandez A , Berenguel M , et al. Adaptive UKF-based model predictive control of a Fresnel collector field[J]. Journal of Process Control, 2019, 85:76-90.
[13] Mahdavi A , Shirazi B , Rezaeian J. Toward a scalable type-2 fuzzy model for resource-constrained project scheduling problem[J]. Applied Soft Computing, 2021, 100(2):106988.
[14] Pimiento N , Serna F . An optimization model to solve the resource constrained project scheduling problem RCPSP in new product development projects[J]. Dyna (Medellin, Colombia), 2020, 87(212):179-188.
[15] Atighehchian A, Sepehri M M, Shadpour P, et al. A two-step stochastic approach for operating rooms scheduling in multi-resource environment[J]. Annals of Operations Research, 2020, 292(4):191-214.
[16] Ghanayem M , Srulovici E , Zlotnick C . Occupational strain and job satisfaction: The job demand-resource moderation_._ediation model in hemodialysis units[J]. Journal of Nursing Management, 2020, 28(1):664-672.
[17] Rezahoseini A, Ghannadpour S F, Hemmati M. A comprehensive mathematical model for resource-constrained multi-objective project portfolio selection and scheduling considering sustainability and projects splitting[J]. Journal of Cleaner Production, 2020, 269:122073.
[18] Rawat P S, Dimri P, Gupta P, et al. Resource provisioning in scalable cloud using bio-inspired artificial neural network model[J]. Applied Soft Computing, 2020, 99:106876.
[19] Fernandez I G, Renjith J A. Resource allocation, scheduling and auto-scaling algorithms for enhancing the performance of cloud using Grey Wolf Optimization and Fuzzy rules[J]. Journal of Intelligent and Fuzzy Systems, 2020(2):1-19.
[20] Boubin M, Shrestha S. Microcontroller Implementation of Support Vector Machine for Detecting Blood Glucose Levels Using Breath Volatile Organic Compounds[J]. Sensors, 2019, 19(10):2283.
[21] Julian J M , Orosco E C , Soria C M . Multi-Sensor Embedded System with Multiple Communications Based on EDU-CIAA[J]. IEEE Latin America Transactions, 2019, 18(2):368-375.
[22] Prauzek M , Konecny J , Vitasek M , et al. Powering Batteryless Embedded Platforms by Piezoelectric Transducers: A Pilot Study[J]. Elektronika ir Elektrotechnika, 2019, 25(2):32-35.
[23] Holovatyy A , Teslyuk V , Kryvinska N , et al. Development of Microcontroller-Based System for Background Radiation Monitoring. 2020, 20(24): 7322-7322.