Scholar Publishing Group
Distributed Processing System
https://doi.org/10.38007/DPS.2022.030204
ISSN 2790-0916 Vol. 3, Issue 2: 42-59

Embedded Real-time Operating System in Remote Viewing of Electrical Automation Equipment

Jumshaid Ullah Khan*

Commune d'Akanda, Gabon

**corresponding author*

Keywords: Embedded Real-time Operating System, Electric Automation, Remote View, Real-time Scheduling Algorithm

Abstract: With the rapid development of computer science and the development of embedded information, digital control systems have huge application development prospects in the field of embedded technology. The chapter mainly discusses the practical application of remote viewing of electrical automation equipment based on embedded real-time operating system. The article first explores the security issues of embedded real-time operating systems and remote viewing systems. It is found that with the development of technology, the possibility of embedded operating system being attacked is increasing, and there is a need for further research on security protection strategies based on embedded systems. Then the embedded real-time scheduling algorithm of PPB-MFT-EDF and embedded real-time schedulable conditions are proposed. Finally, the application of embedded real-time algorithm in micro power supply is explored, which leads to its application effect in electrical automation equipment. The research results show that the energy storage device will charge and discharge low-priced electricity in the valley section, reduce the total power generation cost, and also eliminate the phenomenon of simultaneous charging and discharging of energy storage. Reflecting the superiority of the embedded real-time algorithm, the embedded real-time scheduling algorithm has been reasonably dispatched in the electrical automation equipment, which can improve the performance of the electrical automation equipment by more than 80%.

1. Introduction

With the rapid development of the information technology industry dominated by computers, information communication technology, and microelectronics technology, embedded real-time operating systems are increasingly widely used in aerospace, science and technology, military technology and other fields. In the development process of these applications, except for some simple applications, the rest without exception choose to use the operating system to facilitate the development. In the choice of operating system, a term is gradually becoming known to us-embedded real-time operating system. At the same time, it has become a research hotspot in the computer field, especially the embedded field. Not only that, in the application of electrical automation equipment, embedded real-time operating systems have also begun to be used in order to improve the performance of electrical automation.

Embedded real-time management systems mainly work in areas that require real-time performance. It can effectively manage and adjust the source and operation of the system, which can not only ensure the normal operation of the system, but also enable the control and demand of the system time to be more fully realized. In-depth research on real-time operating systems has also become a key area of embedded system application research and development.

According to the research progress at home and abroad, different scholars have also made corresponding investigations in the embedded real-time operating system: Tamura Y introduced a real-time operating system (RTOS), which provides distributed shared memory (DSM) services for distributed embedded control systems. Model-based design is widely used in embedded control software design. The source code can be generated from the controller model [1]. Archibald J introduced an advanced course to let students understand the basic characteristics of embedded software, such as concurrency, synchronization, and communication. The core of this course is a series of laboratory assignments in which students design and implement real-time operating systems. Each RTOS developed by students has the same API, so they can all run the same application code, but the internal implementation is very different [2]. Xue L studied how to extend the service life of a real-time embedded system (RTES-EH) with energy harvesting capabilities. RTES-EH includes photovoltaic (PV) panels for energy harvesting, super capacitors for energy storage, and real-time sensor nodes as embedded load devices [3]. Alvarez AG described the development and verification of the embedded recognition system (IWR) implemented on the STM32F4-Discovery platform. The system includes a real-time operating system (RTOS) to manage various processing and control tasks. A single file can be used to easily reconfigure different acquisition, processing and identification parameters. The verification is done using the robot control scenario, and the performance that proves the actual usefulness of the system is achieved [4].Perez Tijero H proposed a real-time platform for multi-processor and partitioned systems, which also considered communication requirements, It also describes the adaptation of MaRTE OS (a single-processor real-time operating system) to the XtratuM hypervisor of the multi-processor Intel x86 architecture [5].Wang J has developed a low-cost cloud sensor computing node with embedded Linux operating system, mobile agent middleware and open database functions. The information sharing and exchange is to use mobile agents to distribute analysis algorithms to cloud-aware computing nodes, process data locally and share analysis results [6].Ji Q proposes a fast and accurate real-time video detection and tracking algorithm suitable for embedded systems. The algorithm combines the single-shot multi-box detection target detection model in the deep convolutional network and the core-related filter tracking algorithm. It satisfies the time performance of the actual algorithm on the embedded platform, and solves the problem that the traditional kernel-related filter algorithm cannot perform robust tracking of complex scenes. The experimental results on the embedded platform show that the algorithm can realize real-time tracking of objects in the video, and can automatically reposition the objects to continue tracking after the object tracking fails [7]. Fabricio MA introduces a monitoring system for industrial electrical equipment in production lines, which aims to monitor its operating status in real time, realize machine management and detect deviations and faults early. The system uses sensors connected to a data concentrator module to measure the effective current consumption of the monitored device. The module stores the data collected by these sensors and performs preliminary processing before transmitting to the IoT platform [8]. However, these studies did not introduce the embedded real-time operating system to the application of electrical automation equipment remote viewing, and did not combine the two to illustrate the problem.

The innovations of this article are mainly reflected in:(1) The security issues of embedded real-time operating system and remote viewing system were explored, and it was found that with the development of technology, the possibility of embedded operating system being attacked is also increasing;(2) Proposed based on the embedded real-time scheduling algorithm and schedulable conditions, and then based on the embedded real-time scheduling algorithm to explore the micro power supply and electrical automation equipment.

2. Application Method Based on Embedded Real-Time Operating System in Remote Viewing of Electrical Automation Equipment

2.1. Security Issues of Embedded Real-Time Operating System and Remote Viewing System

After the realization of the electric power remote viewing system network, security problems immediately surfaced. Remote viewing security includes database, video information, control information and system security [9].Due to the particularity of the remote viewing system, the security of the remote viewing system often depends on the ability to accurately identify the personal identity of the user and the terminal, so identity verification has become the prerequisite, cornerstone and core of the security of the remote viewing system [10].

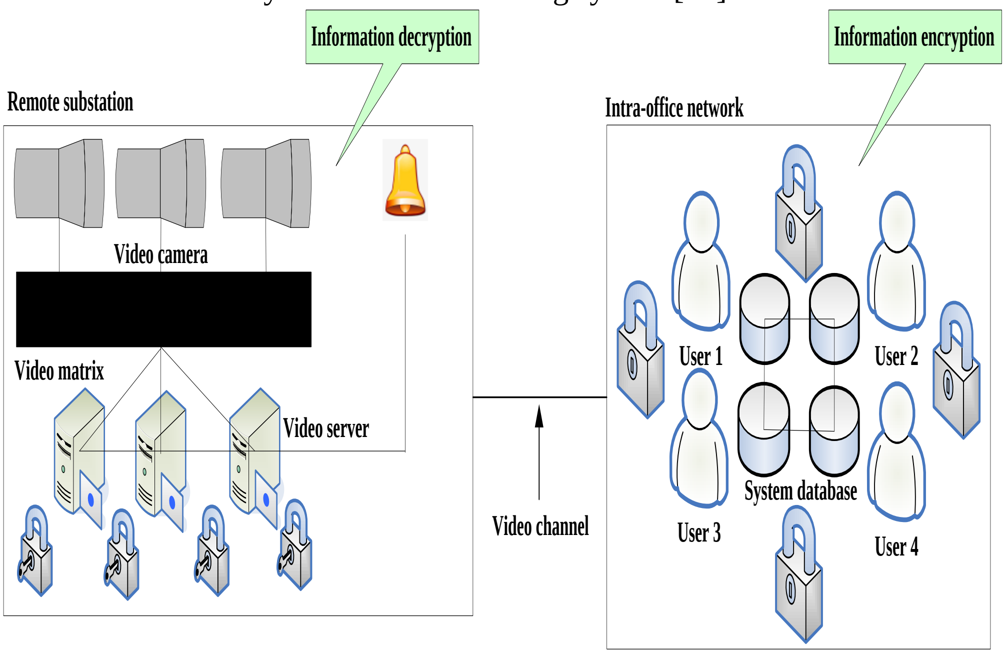

Figure 1. The structure diagram of the remote viewing system after using encryption

In the remote viewing system, as shown in Figure 1, when the control signal sent by the server of the remote viewing system is first encrypted by the preselected button, and then transmitted to the video server of the system through various channels, after the video server receives the encrypted control signal from the server, it is now decrypted, and the agreed key is still used in the decryption process [11].

In terms of network security, the introduction of an embedded real-time operating system is a more likely solution. Due to the differences in the operating mechanism, architecture and operating system of the windows series of embedded systems, ordinary viruses are not easy to infect embedded systems, and it is not easy for hackers to penetrate [12]. Therefore, in new systems or systems that require high system security, embedded network video servers should be used. However, due to the development of technology, the installed operating system is more and more prone to crash, and the security mechanism based on embedded system needs further research [13].

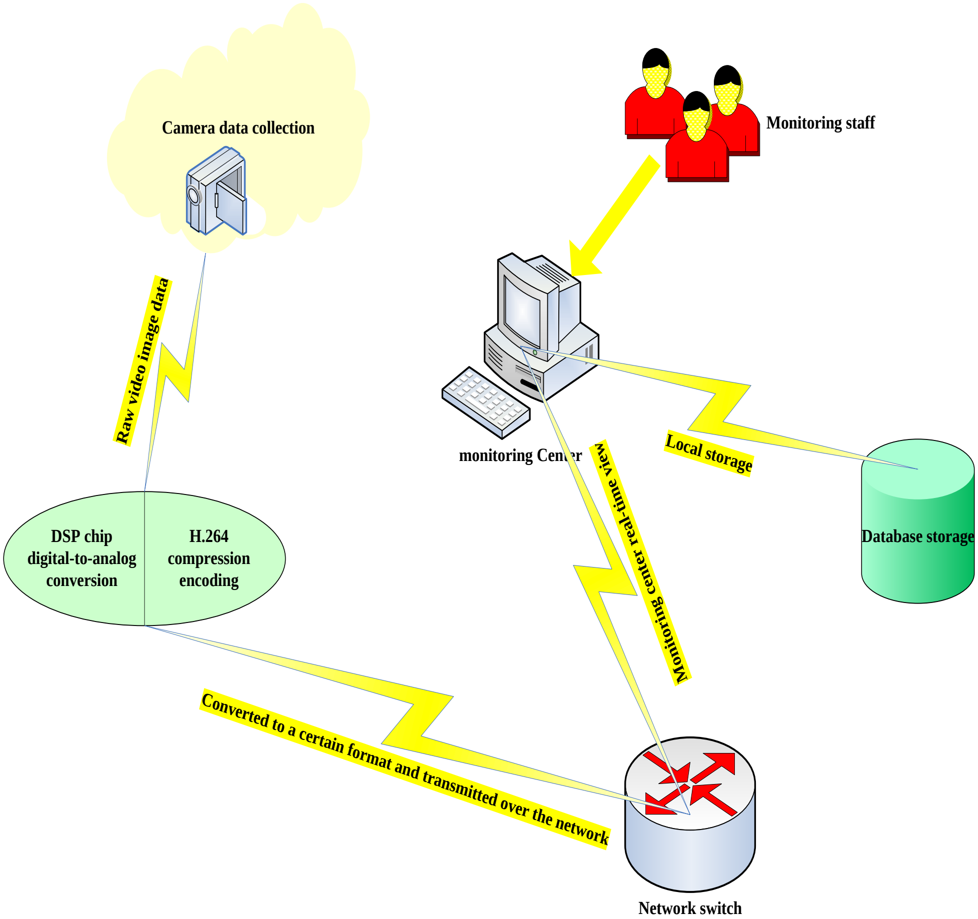

Figure 2. Block diagram of data circulation

The remote viewing control system is established on the original integrated intelligent system of the substation, and on its basis, it involves many advanced technologies. Such as video acquisition technology, data compression technology, data encoding and decoding technology, network communication technology, database storage technology, development software and programming technology, etc.[14].The adoption of each technology involves many aspects of software and hardware, leaving aside the details, the data flow block diagram is shown in Figure 2. Roughly: Unattended substation side monitoring equipment, such as cameras, sensors, alarms, etc., collects the corresponding raw data, after being compressed and encoded it is converted into a format suitable for network transmission, and transmitted to the remote monitoring center through the switch or router in the link. The decompression conversion performed on the monitoring host is displayed on the monitor screen in real time, and the monitoring and operating personnel perform related operations as needed [15].At the same time, the data transmitted from the unattended substation will be converted into the corresponding data format and stored in the local database after a certain format conversion, so that it can be accessed at any time [16].

2.2. Real-Time Scheduling Algorithm Based on Embedded PPB-MFT-EDF

For a given task set $L = \{\mu_1, \mu_2, \cdots, \mu_3\}, m \geq 1$ and processor set $\delta = \{R_1, R_2, \cdots, R_M\}, h \geq 2$, if the parameters meet:

$$\sum_{a=1}^{m} \frac{D_a}{\gamma_a Q} \leq 0.25h \tag{1}$$

Then the task set L is schedulable in the processor set.

Proof: Obtained from the system model: For$\forall \mu_a \in L$, $\mu_a w_r F = \varphi \times \mu_a\, R(0 < \varphi < 1)$ can be assumed. For processor $T_a(1 \leq a \leq h)$, if other processors T_a fail during a certain period of time w, the processor will not only execute the main version task assign to it, At the same time, the first dedicated processor or partition function of the failed processor must be turned off [17]. We assume that the events assigned to each task in the processor can be executed in time, these tasks can be started. Through Baker analysis, it can be concluded that the tasks of all tasks in the processor can be accessed. Conditional deployment includes:

$$\sum_{\mu_l.w^s.T-T_a} \frac{E_l}{\mu_l.w^s.O} + \sum_{\mu_l.w^i.T-T_a} \frac{E_k}{\mu_k.w^i.O} = \sum_{\mu_l.w^s.T-T_a} \frac{E_l}{\varphi\mu_l.O} + \sum_{\mu_l.w^i.T-T_a} \frac{E_k}{(1-\varphi)\mu_l.O} \leq 1 \tag{2}$$

Contains the configuration settings of all the left and right processors at the same time, including:

$$\sum_{a=1}^{h} \left(\sum_{\mu_l.w^s.T-T_a} \frac{E_l}{\varphi\mu_l.O} + \sum_{\mu_l.w^i.T-T_a} \frac{E_k}{(1-\varphi)\mu_l.O}\right) \leq h \tag{3}$$

The resource utilization of the main version task allocated to processor $T_a(1 \leq a \leq h)$ is:

$$\sum_{\mathrm{\mu_l.w^s.T-T_a}} \frac{\mathrm{E_l}}{\mathrm{\mu_l.w^s.O}} = \sum_{\mathrm{\mu_l.w^s.T-T_a}} \frac{\mathrm{E_l}}{\mathrm{\varphi\mu_l.O}} \tag{4}$$

Then the total resource utilization of the major version allocated to all processors is:

$$\sum_{a=1}^{h} \sum_{\mu_l.w^s.T-T_a} \frac{E_l}{\varphi.\mu_l.O} = \sum_{l=1}^{m} \frac{E_l}{\varphi.\mu_l.O} \tag{5}$$

The resource utilization of the minor version task allocated to processor $T_a(1 \leq a \leq h)$ is:

$$\sum_{\mu_l.w^i.T-T_a} \frac{E_k}{\mu_k.w^i.O} = \sum_{\mu_l.w^i.T-T_a} \frac{E_k}{(1-\varphi)\mu_l.O} \tag{6}$$

Then the total resource utilization of the minor version allocated to all processors is:

$$\sum_{a=1}^{h} \sum_{\mu_k.w^i.T-T_a} \frac{E_k}{(1-\varphi)\mu_k.O} = \sum_{l=1}^{m} \frac{E_l}{(1-\varphi)\mu_l.O} \tag{7}$$

Combining (5), (6), (7) has:

$$\sum_{l=1}^{m} \frac{E_l}{\varphi\mu_l.O} + \sum_{k=1}^{m} \frac{E_k}{(1-\varphi)\mu_k.O} \leq h \tag{8}$$

Simplified:

$$\sum_{l=1}^{m} \frac{E_l}{\mu_l.O} \leq h\left[\frac{1}{4}-(\frac{1}{2}-\varphi)^2\right] \tag{9}$$

The left end of the inequality is the utilization rate of the task set. Obviously, when φ takes 0.5, the right end of the inequality has a maximum value of 0.25h, and the proof is complete.

The schedulable and schedulable judgment conditions of the task set given above are too conservative [18]. Figure 3 is a schematic diagram of the algorithm. The task model does not require that the minor version on the processor is schedulable, as long as the major version and the minor version of the base version of the processor can be schedulable [19]. Based on this, the conditions under which the processor can be scheduled are given below.

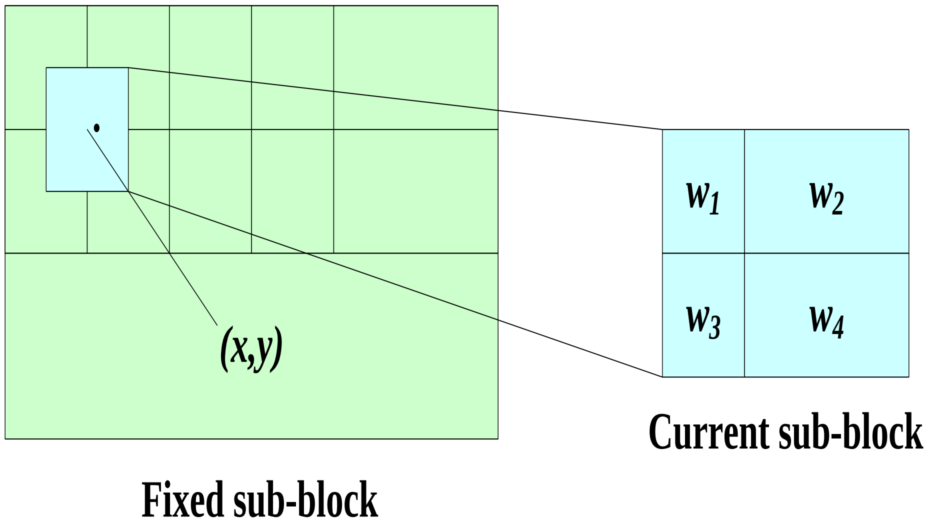

Figure 3. Schematic diagram of the algorithm

The scheduling mode is set to

$$\sum_{k=1}^{n} \frac{E_k}{\mu_k.O} + X_n \leq 0.5 \tag{10}$$

In

$$X_n = max\{\sum_{\grave{\mu_i}.w^i.T\neq T_l} \frac{E_i}{\grave{\mu_i}.O}\} \tag{11}$$

Proof: It can be seen from the previous analysis that only one processor fails in the system at a time. For processor T, assuming that at a certain time w, processor $T_l(T_l \in \tau, k \neq a)$ fails, then processor T_a not only has to perform the basic tasks assign to it, at the same time, it must also execute the base version processor or the sub-version task assigned to it [20]. Based on the scheduling algorithm and failure mode, the time setting between the primary and secondary features is much shorter than the operating cycle. And through Baker's analysis, it can be concluded that the conditions for programming tasks in the processor are:

$$\sum_{k-1}^{n} \frac{E_k}{0.5\mu_k.O} + \sum_{\grave{\mu_i}.w^i.T\neq T_l} \frac{E_k}{0.5\mu_k.O} \leq 1 \tag{12}$$

Because T_l is arbitrary. Obviously when it is the maximum value for all processors (except T_a). Processor T_a is in the worst case. If the above formula is satisfied at this time, the task is schedulable [21]. Then the tasks on processor T_a can be scheduled under any circumstances. The conclusion is established. Table 1 shows the characteristics of the processor.

Table 1. Features of the processor

Characteristic		Processor core		
		Nios II/f	Nios II/e	Nios II/s
Performance	DMIPS/MHz	1.21	0.21	0.69
	Max DMIPS	221	29	131
	Maximum operating frequency	185 MHz	200 MHz	165 MHz
Pipeline order		7	2	6
External addressing space		2G bytes	2G bytes	2G bytes
Hardware multiplier		1 cycle	Software simulation verification	3 cycles
High-speed cache		512-64K bytes	without	512-64K bytes
Branch prediction		dynamic	without	Static
User mode support		no	no	no
Custom instruction support		237	237	237

If the EDF algorithm is used for the local scheduling of the processor, the minimum number of processors required for task set deployment is:

$$h_{min} \geq 2\sum_{a=1}^{m}\frac{E_a}{\mu_a.O}+1 \tag{13}$$

Proof: Assume that the first part of each task is assigned to each task according to the task; for a major version task on processorT_a, their minor versions are also evenly distributed to other processors according to the utilization rate. The resource utilization rate of each processor (using the ratio of execution time to time limit) is:

$$\frac{\sum_{a=1}^{m}\frac{E_a}{0.5\mu_a.O}}{h}\left(1+\frac{1}{h-1}\right)=\frac{\sum_{a=1}^{m}\frac{E_a}{0.5\mu_a.O}}{h-1}=X_{o_1} \tag{14}$$

The above formula can be simplified as:

$$(h-1)X_{o_1}=2\sum_{a=1}^{m}\frac{E_a}{\mu_a.O} \tag{15}$$

It can be known from this formula that when $\sum_{a=1}^{m}\frac{E_a}{\mu_a.O}$ is constant, X_{o_i} is larger and the number of processors h is smaller.When using the EDF algorithm to program the local system of the processor, the maximum source utilization of the processor can be 1, and when the number of processes required is the least, it is $\sum_{k=1}^{m}\frac{E_k}{\mu_k.O}$ then: $h_{min} \geq 2\sum_{a=1}^{m}\frac{E_a}{\mu_a.O}+1$, the certification is complete.

If the task resource utilization ratio adopts the execution time and cycle ratio, the maximum value of X_{o_1} is 0.5, and the conclusion still holds.

From the above proof process, it can be obtained: Make the resource utilization of each processing process balance and try to achieve the maximum value, so that the number of processing processes can reach the lowest value or the task rejection rate is the lowest [22].

Among them, the relationship between the number of micro-processors required by the system and the efficiency of system resources is relative to the task rejection rate under a fixed processor set, as shown in Figure 4:

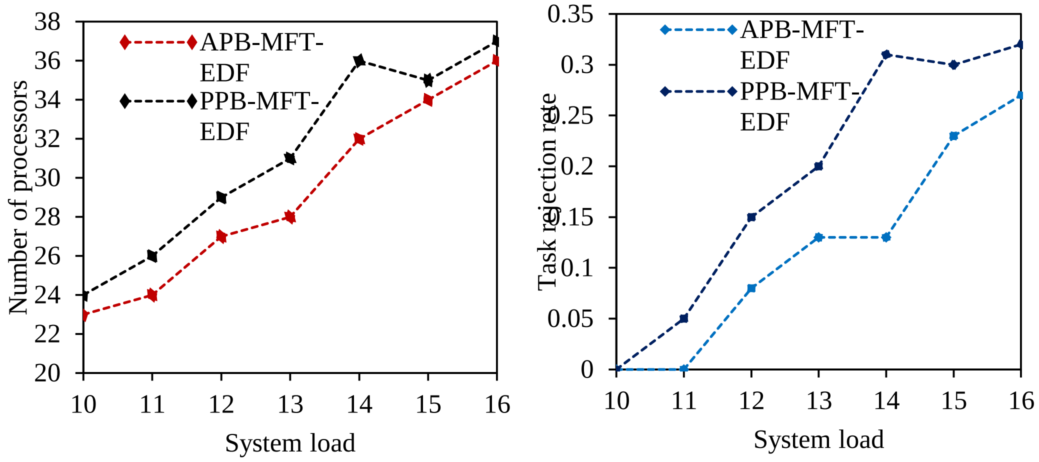

Figure 4. Comparison of the number of processors required by the APB-MFT-EDF algorithm and the PPB-MFT-EDF algorithm and comparison of the task rejection rate

It can be found that the resource scheduling algorithm using active backup fault-tolerant technology is better than the algorithm of passive backup fault-tolerant technology. This is mainly because the passive backup fault tolerance rate technology strictly divides it by controlling the different start date and time between the main version and the minor version. Although their running time is not repeated, this will increase the resource utilization of the task by several times; In the active backup fault-tolerant rate technology, although the running time of the primary and secondary versions is repeated, but especially when there is a repetitive running time instead of a processor failure corresponding to a version, the normal operation of some tasks will be invalid, and the task resource efficiency is not as good as the passive backup method [23].

2.3 Embedded Real-Time Schedulable Conditions

Suppose the real-time task of the base version assigned to processor T_a is $L_o = \{\mu_1, \mu_2, \cdots \mu_n\}$, and the real-time task of the minor version assigned to it is $L_a = \{\grave{\mu_1}, \grave{\mu_2}, \cdots, \grave{\mu_n}\}$.

Processor set:

$$\tau = \{T_1, T_2, \cdots, T_h\}, h \geq 2 \tag{16}$$

If the parameters meet:

$$\sum_{a=1}^{m} \frac{E_a}{\mu_a.O} \leq 0.5h \tag{17}$$

Then task set L can be scheduled in processor set μ.

Proof: For processor$T_a(1 \leq a \leq h)$, suppose that at a certain time w, processor T_a will not only execute the main version task assigned to it, but also execute the sub-version task assigned to it at the same time [24].Assuming that the instances assigned to each task (including the major version and the minor version) in a single processor can be implemented within its time, then these tasks are adjustable, through Baker's analysis, it can be concluded that the conditions for task allocation in a single processor are:

$$\sum_{k=1}^{n} \frac{E_k}{\mu_a.O} + \sum_{a=1}^{i} \frac{E_a}{\mu_a.O} \leq 1 \tag{18}$$

The configuration settings of all processors are merged at the same time, there are:

$$\sum_{a=1}^{h}(\sum_{k=1}^{n}\frac{E_k}{\mu_a.O}+\sum_{a=1}^{i}\frac{E_a}{\mu_a.O})\le h \quad (19)$$

Obtained from the above formula:

$$\sum_{l=1}^{m}\frac{E_l}{\mu_l.O}+\sum_{k=1}^{m}\frac{E_k}{\mu_k.O}\le h \quad (20)$$

Then the above formula can be simplified to:

$$\sum_{\mathrm{a}=1}^{\mathrm{m}}\frac{\mathrm{E_a}}{\mathrm{\mu_a.O}}\le 0.5\mathrm{h} \quad (21)$$

So far it proves that the conclusion is established.

An important property of the algorithm is given below:

The data processor partially uses the EDF adjustment algorithm, and the minimum number of information processors required for the task set to be schedulable is about:

$$h_{min}\ge [2\sum_{a=1}^{m}\frac{E_a}{\mu_a.O}] \quad (22)$$

Proof: Assume that the first part of the task is allocated to each processor according to the utilization rate; For the major version tasks on a processor T_a, their minor versions are equally distributed to other processors $T_l(l\ne a)$ according to the utilization rate, then the resource utilization rate of each processor is:

$$\sum_{k=1}^{n}\frac{E_k}{\mu_a.O}+\sum_{a=1}^{i}\frac{E_a}{\mu_a.O}=\frac{2}{h}\sum_{a=1}^{m}\frac{E_k}{\mu_a.O}=X_o \quad (23)$$

From the above formula, we can get:

$$hX_o=2\sum_{a=1}^{m}\frac{E_k}{\mu_a.O} \quad (24)$$

It can be seen from the above formula that when $\sum_{a=1}^{m}\frac{E_k}{\mu_a.O}$. When it is determined, X_o is the highest, and the number of processors h used is the smallest. When the processor adopts the EDF part of algorithm design, the maximum resource utilization of the processor can be 1, and the number of processes required at this time is the least. Namely:

$$h_{min}=[2\sum_{a=1}^{m}\frac{E_a}{\mu_a.O}] \quad (25)$$

The results can be obtained from the above proof process: the resource utilization of each processor is balanced and maximized, and the number of processors can also be minimized or the number of rejected tasks can be minimized [25].

Like general non-embedded programs, before running in a multitasking environment, an entry function program will be executed first after the system is powered on, that is, the usual main function. This main function mainly completes the initialization of some applications and operating systems, and allows the system to enter the multi-task operating environment managed by the real-time operating system. Its program processing flow is shown as in Fig 5.

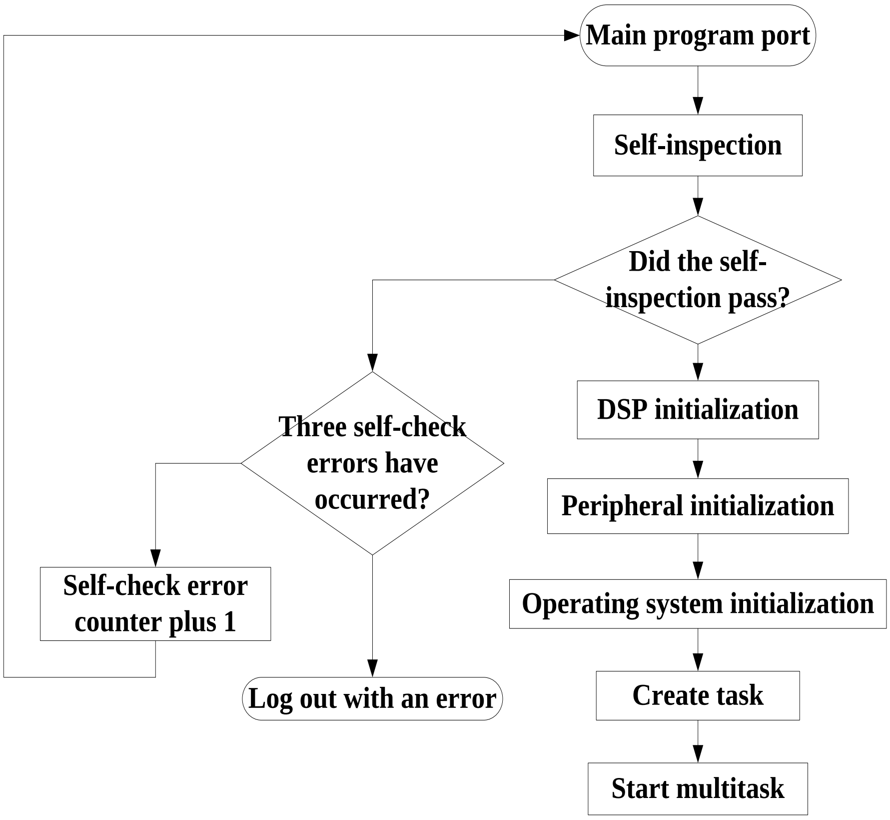

Figure 5. The main function program flow chart

3. Application Experiment Results Based on Embedded Real-Time Operating System in Remote Viewing of Electrical Automation Equipment

3.1. Embedded Real-Time Algorithm in Micro Power Supply

The most basic unit in solar photovoltaic power generation is photovoltaic cells, which are energy converters that convert light energy into electricity. Its basic principle is to project sunlight directly onto the semiconductor components in the photovoltaic cell to form a photoelectric effect, which can be operated on a network or independently. At present, because the technology of solar photovoltaic power generation is not mature yet, its operating cost is too high, and it is as uncompetitive as traditional power generation technology. However, its application prospects are generally recognized by scientific workers because of its low operating cost, no pollution to the environment, and its own maintenance is also very convenient. Because the output power of a single solar photovoltaic cell is relatively small when generating electricity, several solar photovoltaic power cells are connected in series and parallel to form a solar photovoltaic module system. The output voltage of this module can reach dozens of volts or more, and then multiple photovoltaic modules can be connected in series and parallel to obtain a photovoltaic array. This array is a photovoltaic power generation system, and its output power is greater.

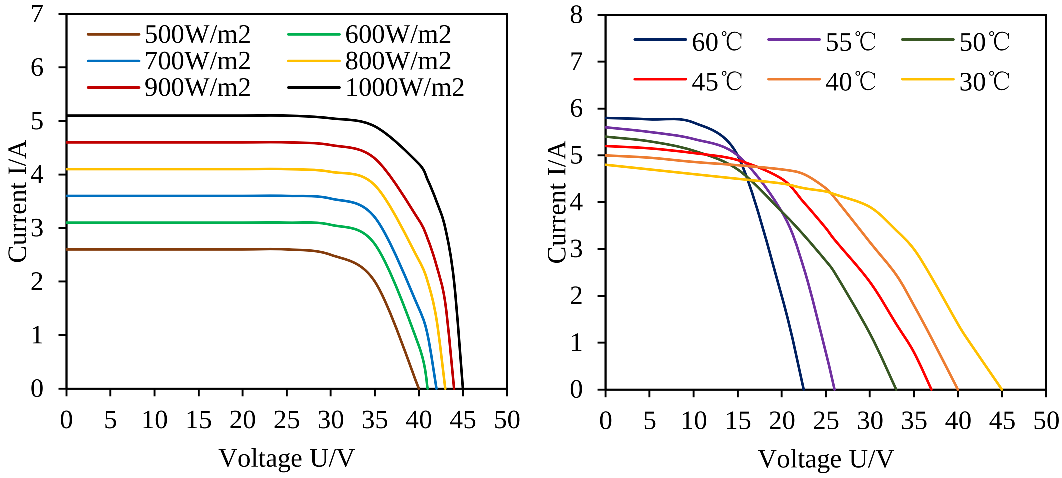

Figure 6. The volt-ampere characteristics of the photovoltaic cell

Figure 6 shows the volt-ampere characteristic curve of the photovoltaic cell. Because of factors such as ambient temperature, lighting intensity, load conditions, or cloudy, sunny, rainy, and foggy, photovoltaic cells produce non-linear characteristics of output power. If the lighting intensity and temperature change, the output power will change accordingly. With a certain output current, the battery's power generation output can reach the maximum. Relative to the apex of the current curve, this highest point is also called the maximum power point. Long term work of the photovoltaic cell around the maximum power point, the best power generation effect, this is the maximum power point tracking. Normally, photovoltaic cells are operated in MPPT mode.

In the isolated network state, all the power required by the user is provided by the micro power supply. After using an improved real-time algorithm to optimize, the output power of each micro power supply varies with time, as shown in Figure 7.

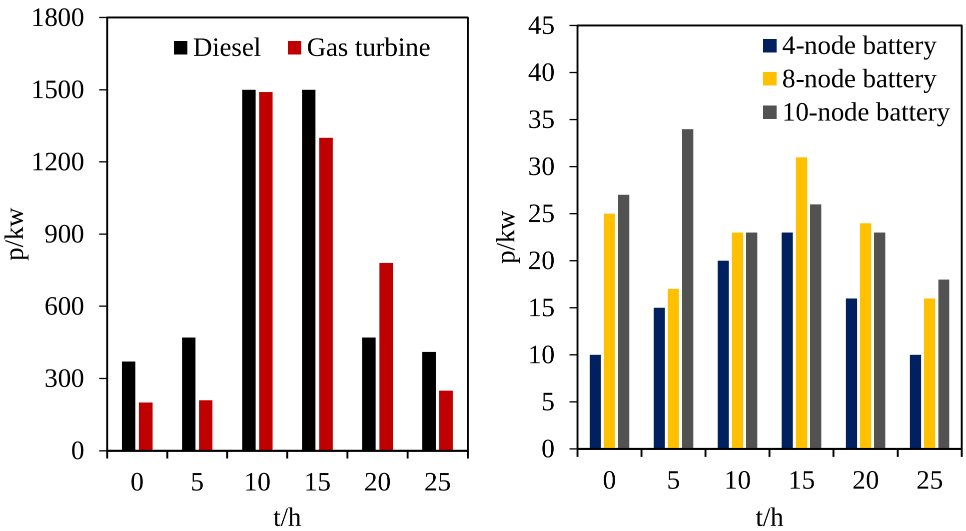

Figure 7. The output power of each micro power supply after optimization of the improved real-time algorithm

After using embedded and real-time algorithm optimization, the production pressure of each micro power supply changes with time as shown in Figure 8.

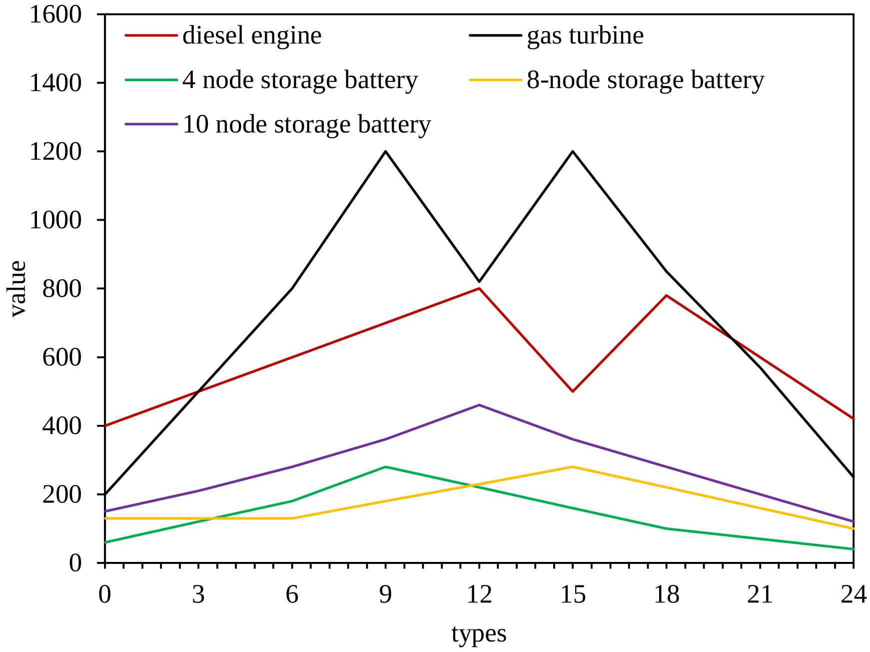

Figure 8. Embedded and real-time algorithm output power of each micro power supply

For Figure 7, all three batteries should have been charged in the 10th period. However, because the electricity generated is not enough to satisfy the user and the remaining 3 batteries can be charged, only 2 batteries can be charged, so the third battery is not charged. When the eleventh period is discharged, the two charged batteries in the 10th period begin to discharge. Since the cost of battery discharge at this time is lower than the cost of power generation of diesel engines and gas turbines, one of the batteries is discharged and charged into the battery that has not been charged in 10 hours. In this way, the phenomenon of simultaneous charging and discharging is formed, and finally all the energy of all batteries is in the 12th period.

The advantage of Figure 8 relative to Figure 7 is that each energy storage device does its best to charge during the valley period, and the energy storage device discharges the charged electricity during the peak period, because the electricity consumption is low during the valley period. The efficiency of gas turbines is low, and the cost of power generation is higher than that of diesel engines. Therefore, diesel engines try their best to generate electricity and store energy to charge as much as possible. During peak periods, the unit power generation cost of diesel engines and diesel engines becomes higher. The efficiency of gas turbines has become higher, and the cost of power generation is relatively low. It embodies the superiority of embedded and real-time algorithms.

The study found that in the grid-connected operation mode, the total load power of the system is provided by the micro power sources and the main network in the network, and the power demand of the load can be met at any time. In the planning and dispatching layer, the economy of the microgrid is mainly considered. In the real-time dispatching layer, in addition to the economy, the influence of the power error caused by the volatility of the uncontrollable power generation source on the microgrid is also considered. Figure 9 is the output curve of the controllable micro power sources DE, MT, and FC in the planning layer and dispatching layer in the microgrid when it is connected to the grid. The left picture is the power curve of the controllable micro power supply at the planning level, and the right picture is the power curve of the controllable micro power supply at the real-time level.

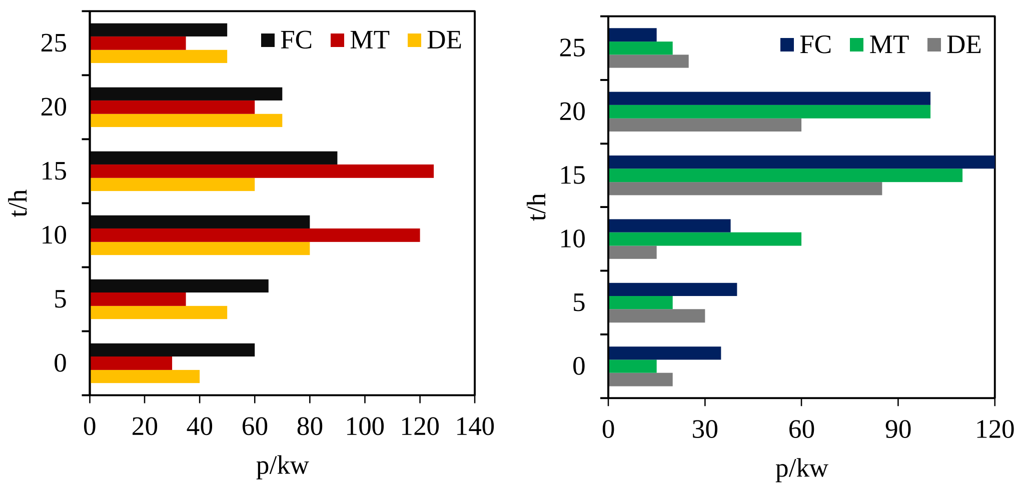

Figure 9. The output curves of the controllable micro power sources DE, MT, and FC at the planning and dispatching levels in the microgrid when grid-connected

In the planning layer and the real-time layer, there are also fluctuations in the conversion power of the main network, the charge and discharge power of the battery and its energy. The curve is shown in Figure 10.

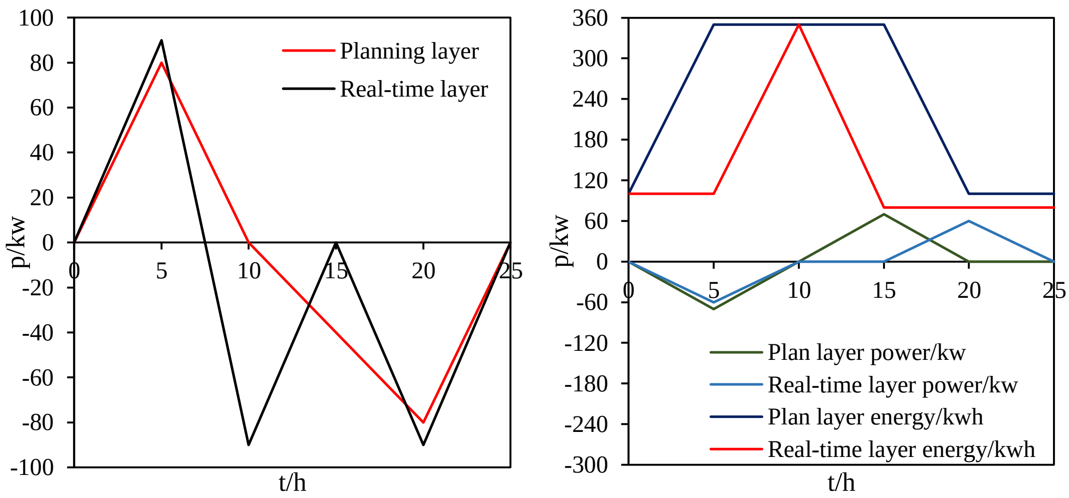

Figure 10. Charging and discharging power and energy curve

The left picture is the energy exchange curve between the planning layer and the dispatching layer and the main network, and the right picture is the charging and discharging and energy curves of the storage battery of the planning layer and the dispatching layer.

3.2. Application of Embedded Real-Time Operating System in Electrical Automation Equipment

In this section of the experiment, the main consideration is the completion rate of the embedded real-time operating system in the electrical automation device, periodic tasks and non-periodic tasks. And the turnaround time limit of non-periodic tasks and the turnaround time limit of authorized tasks. For the non-periodic tasks that have not been realized, the general rotation time limit and the weighted rotation time limit are recorded as zero, and the statistics of the general rotation time limit and the general weighted rotation time limit are not included. The closer the weighted rotation time limit is to 1, the faster the response time limit for non-periodic tasks.

Theoretical data experiment: Suppose there are three periodic tasks and three non-periodic task sets. The mixed task set is scheduled through embedded real-time algorithms, and the completion rates of periodic tasks and non-periodic tasks are shown in Table 2.The results of aperiodic task scheduling are shown in Table 3.

Table 2. Completion rate of experimental mixed tasks

Task set	Number of tasks completed	Total number of tasks	Task completion rate
Task 1	11	14	91.98%
Task 2	9	9	100%
Task 3	8	8	100%
Non-periodic task	4	4	100%

Table 3. Experimental non-periodic task scheduling results

Task set	Time of arrival	Service hours	Complete time	Turnaround time	Weighted turnaround time
Task a	19	9	69	49	4
Task b	29	28	59	27	2
Task c	149	2	147	2	3
Average time	-	-	-	26	3

From the periodic task scheduling results in Table 2, it can be seen that the three non-periodic tasks are all completed, so most of the task completion rate can be higher guaranteed. It can be seen that none of the non-periodic tasks are scheduled at the last moment of their deadlines, and the average weighted turnaround time is only 2.33, which is completely acceptable. Among them, the weighted turnaround time of task 2 and task 3 even reaches 1, indicating that the time when the non-periodic task arrives is the time when it is scheduled, and the response speed is fast, which satisfies the real-time nature of the task.

Simulation data experiment: Suppose there are 5 periodic tasks and 5 non-periodic tasks in the work plan. The amount of time and running time are the number of markers based on a given time. Assume that the total running time is 5000 times units. The mixed work schedule composed of periodic and irregular tasks is shown in Table 4, with a workload of 83%.Among them, type 1 means cyclical activity, and 0 means non-cyclical activity. The embedded real-time algorithm scheduling results are shown in Table 5.

Table 4. Experimental task set

Task set	Cycle/time to reach	Operation hours	Deadline	Task type
Task 1	179	31	-	1
Task 2	171	30	-	1
Task 3	209	33	-	1
Task 4	90	16	-	1
Task 5	190	29	-	1
Task a	101	3	159	0
Task b	79	10	179	0
Task c	100	16	189	0
Task d	70	15	158	0
Task e	91	9	159	0

Table 5. Experimental scheduling results

Periodic task set	Task completion rate	Aperiodic task set	Weighted turnaround time
Task 1	100%	Task a	0.9999
Task 2	100%	Task b	1.0983
Task 3	100%	Task c	4.6978
Task 4	100%	Task d	1.1929
Task 5	100%	Task e	1.4701

It can be found from the test results in Table 5 that all periodic tasks can achieve 100% times scheduling and have a high completion rate. Among the non-periodic tasks, only the average weighted turnaround time of task c is greater than 4, and the average weighted turnaround time of other tasks is less than 1.5. The average weighted turnaround time of the task set is about 1.8921. The response time is short, all five non-periodic tasks are realized, and most of the system resources are effectively used.

From the above two sets of experiments, it can be seen that the embedded real-time scheduling algorithm has been reasonably scheduled in electrical automation equipment, which can improve the performance of electrical automation equipment by more than 80%.

4. Discuss

Energy is an important foundation for the survival and development of the human world, the main support of contemporary human economic society and the development of the world economy, and a powerful indicator to measure the overall energy balance of a country. However, in recent years, due to the rapid development of the global financial system, the demand for energy has continued to grow. The excessive use of fossil fuels such as crude oil and coal combined with the decline in the development of nuclear power technology has caused a global energy supply crisis.

At the same time, it has also caused very serious environmental problems, so reducing the energy crisis is the fastest case in the world. This requires the development and utilization of renewable energy and the construction of a sustainable energy system have become the consensus and tradition of all developed countries in the world today, and more attention should be paid to the development and utilization of renewable energy. With the rapid development and growth of social economy, energy cannot meet the needs of all walks of life, and our dependence on foreign energy markets is becoming more and more important. The heavy snow in the south at the beginning of the 21st century hindered the normal operation of the power grid and even caused the power grid to almost collapse. This in turn weakened road work, hindered coal mining, and triggered a major crisis in the energy system. This shows the weakness of the current power supply system. Once it encounters a large area of harsh environment, it will lead to power shortages. At the same time, we should also actively explore new embedded real-time operating systems so that they can play a better advantage in electrical automation equipment and save energy.

5. Conclusion

With the further improvement of the modern technology level of power system automation and management, in order to intuitively, accurately, quickly and intelligently understand the working conditions and environmental conditions of power system equipment, power remote viewing systems and embedded real-time operating systems have become more and more popular in power systems. And it is moving towards digitalization, networking, integration with SCADA and EMS systems, and automated image recognition. These technologies are integrated into the traditional SCADA, EMS and release security management and control system, of course, will produce major changes to the traditional power system, and may become a new EMS multimedia system. Although the general platform of power system automation based on embedded real-time operating system in this paper has made progress, there are still many problems that need to be further studied. The analysis of other materials in the power system, such as the automatic identification and analysis of infrared images of inverters and circuit breakers, has not yet been involved. Accurately identifying these devices is also of great significance to the safe operation of the power system.

Funding

This article is not supported by any foundation.

Data Availability

Data sharing is not applicable to this article as no new data were created or analysed in this study.

Conflict of Interest

The author states that this article has no conflict of interest.

References

[1] Tamura Y , Doan T T , Chiba T , et al. A real-time operating system supporting distributed shared memory for embedded control systems. Cluster Computing, 2017, 13(10):1-10.

https://doi.org/10.1007/s10586-017-1140-9
[2] Archibald J , Wilde D . Embedded software education: an RTOS-based approach. Acm Sigbed Review, 2017, 14(1):71-80. https://doi.org/10.1145/3036686.3036695
[3] Xue L , Wang Y , Chang N , et al. Concurrent Task Scheduling and Dynamic Voltage and Frequency Scaling in a Real-Time Embedded System With Energy Harvesting. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2016, 35(11):1890-1902. https://doi.org/10.1109/TCAD.2016.2523450
[4] Alvarez A G , Evin D A , Verrastro S . Implementation of a Speech Recognition System in a DSC. IEEE Latin America Transactions, 2016, 14(6):2657-2662. https://doi.org/10.1109/TLA.2016.7555234
[5] Perez Tijero H , Rivas M A , Ortega D M . Multiprocessor platform for partitioned real-time systems: Multiprocessor platform for partitioned real-time systems. Software Practice and Experience, 2016, 47(1):61-78. https://doi.org/10.1002/spe.2404
[6] Wang J , Zhang L , Duan L , et al. A new paradigm of cloud-based predictive maintenance for intelligent manufacturing. Journal of Intelligent Manufacturing, 2017, 28(5):1125-1137.
[7] Ji Q , Dai C , Hou C , et al. Real-time embedded object detection and tracking system in Zynq SoC. EURASIP Journal on Image and Video Processing, 2021, 2021(1):1-16.
[8] Fabricio M A , Behrens F H , Bianchini D . Monitoring of Industrial Electrical Equipment using IoT. IEEE Latin America Transactions, 2020, 18(8):1425-1432. https://doi.org/10.1109/TLA.2020.9111678
[9] Kim B , H Yang. Reliability Optimization of Real-Time Satellite Embedded System Under Temperature Variations. IEEE Access, 2020, 8(99):1-1.
[10] Tsai C Y , Liao H C , Hsu K J . Real-time embedded implementation of robust speed-limit sign recognition using a novel centroid-to-contour description method. Iet Computer Vision, 2017, 11(6):407-414. https://doi.org/10.1049/iet-cvi.2016.0082
[11] Almeida R , Manuel C . System Support and Dependable Adaptation in Small Real-Time Embedded Systems. IEEE Latin America Transactions, 2016, 14(10):4402-4408. https://doi.org/10.1109/TLA.2016.7786322
[12] Barladian B K , Shapiro L Z , Mallachiev K A , et al. Visualization Component for the Aircraft Real-Time Operating System JetOS. Programming and Computer Software, 2020, 46(3):167-175.
[13] Baskiyar S , Huang C C , Tam T Y . Minimum energy consumption for rate monotonic scheduled tasks. Computing, 2016, 98(6):661-684. https://doi.org/10.1007/s00607-015-0475-4
[14] Cvm A , Vh B . A Host Intrusion Detection System architecture for embedded industrial devices. Journal of the Franklin Institute, 2021, 358(1):210-236.
[15] Choi H , Koo Y , Park S . Quantitative Analysis of Power Consumption for Low Power Embedded System by Types of Memory in Program Execution. Journal of Korea Multimedia Society, 2016, 19(7):1179-1187.
[16] Rathai K , Alamir M , Sename O , et al. A Parameterized NMPC Scheme for Embedded Control of Semi-active Suspension System - ScienceDirect. IFAC-PapersOnLine, 2018, 51(20):301-306. https://doi.org/10.1016/j.ifacol.2018.11.029
[17] Wang J , Fu Y . Design and implementation of scintillation index measuring system based on ARM9. Optik - International Journal for Light and Electron Optics, 2016, 127(6):3171-3174. https://doi.org/10.1016/j.ijleo.2015.12.071
[18] Emelenko A N , Mallachiev K A , Pakulin N V . Developing a Debugger for Real-Time Operating System. Proceedings of the Institute for System Programming of RAS, 2016,

28(2):193-204.
[19] Kafadar O , Sertcelik I . A Computer-Aided Data Acquisition System for Multichannel Seismic Monitoring and Recording. IEEE Sensors Journal, 2016, 16(18):6866-6873.
[20] Lee Y , Waterman A , Cook H , et al. An Agile Approach to Building RISC-V Microprocessors. IEEE Micro, 2016, 36(2):8-20. https://doi.org/10.1109/MM.2016.11
[21] Dabove P , Pietra V D . Towards high accuracy GNSS real-time positioning with smartphones. Advances in Space Research, 2019, 63(1):94-102.
[22] Sewell T , Kam F , Heiser G . High-assurance timing analysis for a high-assurance real-time operating system. Real-Time Systems, 2017, 53(5):812-853.
[23] Donghwan, Son, Hwayooung, et al. Highly reliable partitioning real-time operating system and case studies on weapon system applications. Communications of the Korean Institute of Information Scientists and Engineers, 2016, 34(10):53-59.
[24] Greeff H , Manandhar A , Thomson P , et al. Distributed Inference Condition Monitoring System for Rural Infrastructure in the Developing World. IEEE Sensors Journal, 2019, 19(5):1820-1828. https://doi.org/10.1109/JSEN.2018.2882866
[25] Ovatman T , Aral A , D Polat, et al. An overview of model checking practices on verification of PLC software. Software & Systems Modeling, 2016, 15(4):937-960. https://doi.org/10.1007/s10270-014-0448-7