Energy consumption is an important indicator for evaluating soft real-time periodic system performance. This work is based on previous lab work of processing of single task. It aims to reduce energy consumption for periodic soft real-time systems with multi-tasks and multiple deadlines, by using Dynamic Voltage and Frequency Scaling (DVFS) and reinforcement learning to learn the complex task pattern, and then adjust the frequency based on the system environment. The method combines DVFS with reinforcement learning. DVFS is a commonly used method in the Linux kernel to reduce energy consumption where the DVFS governor in Linux decides how to adjust the frequency of the next time period according to the current load of the system, in order to save energy dynamically. The reinforcement learning method used in this work is Deep Recurrent Q-Network (DRQN). The general process of this method is to run a set multi-task multi-deadline workload. In the Linux kernel, neural networks are embedded inside the governor to compute the Q-value of each selectable frequency and the Q-value is used to select the next frequency. The embedded neural network consists of two parts, a layer of gated recurrent units (GRU) and a multi-layer perceptron (MLP). The GRU output, along with other parameters are inputted into the MLP together. The frequency will also be put into MLP. Based on the Q value calculated by the MLP, the governor selects the frequency for the next time period. Each time the governor makes a decision, the inputs and outputs of the neural network, as well as other processing signals, are saved. After the whole workload is finished, this information is read by the user space to train the neural network with DRQN and update the parameters of the neural network. The updated parameters are sent into the kernel to continue running the workload for the next cycle. This process is repeated hundreds of times to learn a better neural network to save energy. A simple workload with 3 tasks and 3 deadlines is currently being tested for this system. The same CPU-intensive benchmark is started at times 0, 0.3, and 0.7, with corresponding deadlines of 0.3, 0.4, and 0.5. The workload was tested and found that the system can already recognize different deadlines and adjust the frequency so that each task is closer to its own deadline to save energy.