VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Juho Vanhamäki

PROTECTION RELAY SELF-SUPERVISION STUDY

Technology and Communication
2021

VAASAN AMMATTIKORKEAKOULU
Tietotekniikka

TIIVISTELMÄ

Tekijä	Juho Vanhamäki
Opinnäytetyön nimi	Protection Relay Self-supervision Study
Vuosi	2021
Kieli	englanti
Sivumäärä	25
Ohjaaja	Jani Ahvonen

Tämän opinnäytetyön tavoitteena oli tutkia erään suojareleen itsevalvontaa ja sen toteutusta. Tähän työhön ei kuulunut muutosten tekeminen itsevalvontaan, vaan vain sen toteutuksen tutkiminen ja dokumentointi. Työ voidaan jakaa kolmeen eri kokonaisuuteen:

Ensimmäisessä osassa tutkitaan sulautettujen laitteiden itsevalvonnan periaatteita ja eri suojarelevalmistajien dokumentaatiota suojareleensä itsevalvonnasta. Tutkittu dokumentaatio on julkista ja löytynyt valmistajan julkisilta verkkosivuilta. Tuotteet on valittu sen perusteella, että ne ovat mahdollisimman samankaltaisia tutkimuksen aiheena olevan suojareleen kanssa.

Toinen osa keskittyy tämänhetkiseen työn aiheena olevan suojareleen itsevalvonnan toteutukseen. Yleiset periaatteet, itsevalvonnan eri tilat ja toimenpiteet on tutkittuna. Jotkin vikakoodit on avattu tarkemmin. Tutkimuksessa on käytetty suojareleen ohjelmakoodia siinä muodossa missä se oli maaliskuussa 2021.

Viimeinen osa keskittyy itsevalvonnan tulevaisuuteen ja ideoihin sen toteutuksen muuttamisesta. Nämä muutokset käydään läpi vain ajatuksen tasolla, eikä niiden mahdollinen toteutuminen tulevaisuudessa kuulu tämän opinnäytetyön alueeseen.

Avainsanat Suojarele, ABB, itsevalvonta, IRF

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikka

ABSTRACT

Author	Juho Vanhamäki
Title	Protection Relay Self-supervision Study
Year	2021
Language	English
Pages	25
Name of Supervisor	Jani Ahvonen

The aim of this thesis was to study the self-supervision of a protection relay. This thesis does not include making any changes in the self-supervision, only to study the current implementation and document it. This thesis can be separated to three general parts:

The first part of this thesis focused on general principals of embedded device self-supervision and a comparison of self-supervision documentation from different manufacturers of protection relays. This is done based on free documentation found on the manufacturers' websites. The products were chosen based on their similarity to protection relay being studied.

The second part focused on the current implementation of self-supervision. General principals, self-supervision states and actions were studied. Some IRF codes of interest were also given a closer look. The study was done on protection relay code base as it was in March 2021.

The final part focused on future of self-supervision and ideas of changes in its implementation. These changes are only discussed on idea-level and their possible implementation is not part of this thesis.

Keywords Protection relay, ABB, self-supervision, IRF

TABLE OF CONTENTS

TIIVISTELMÄ

ABSTRACT

1 INTRODUCTION 7

2 GENERAL IDEAS OF EMBEDDED DEVICE SELF-SUPERVISION 8

2.1 Power-On Self Tests (POST) 9

2.1.1 CPU, Register, Interrupt and Exception Test 9

2.1.2 RAM test 9

2.1.3 Extension module/device/card/peripheral -existence test 11

2.2 Runtime tests 11

2.2.1 Trip output tests 11

2.2.2 Measurement functionality supervision 12

2.2.3 Runtime RAM test 12

2.2.4 Runtime extension module existence test 13

2.3 Watchdog 13

2.3.1 Software watchdog 13

2.3.2 External watchdog 14

2.4 Self-supervision actions when fault detected 15

2.4.1 During POST 15

2.4.2 During runtime 15

3 COMPARISON OF PROTECTION RELAY SELF-SUPERVISION DOCUMENTATION 17

3.1 ABB 17

3.2 Arcteq 18

3.3 Siemens 19

3.4 Schneider-VAMP 21

4 CONCLUSIONS 23

REFERENCES 24

LIST OF FIGURES AND TABLES

Figure 1. Maxim Integrated MAX823 external watchdog component. 14

Figure 2. ABB feeder protection relay REF615. 17

Figure 3. Arcteq feeder protection relay AQ-F215. 19

Figure 4. Siemens Reyrolle-series protection relay 7SR51. 20

Figure 5. VAMP 321 arc-protection relay. 21

GLOSSARY

SEU	Single-event upset
IRF	Internal relay fault
ISR	Interrupt Service Routine
PSM	Power Supply Module
BIO	Binary input/output
CPU	Central Processing Unit
FPGA	Field-Programmable Gate Array
RAM	Random-Access Memory
(L)HMI	(Local) Human-Machine Interface
POST	Power On Self Test(s)
EEPROM	Electronically Erasable Programmable Read-Only Memory
ADC	Analog-to-Digital Converter

1 INTRODUCTION

This thesis was made for ABB Oy, Distribution Solution unit. It is made to study the self-supervision of a protection relay.

Most modern protection relays have at least some forms of self-supervision. In case of an inner fault of the device, the self-supervision activates and initiates pre-defined actions. These actions can vary from info messages on the device screen to full lockdown of the device.

The aim of this thesis was to study the current self-supervision implementation of a protection relay, to understand its current functionalities and actions. This document can then be used as a reference in the future when self-supervision of a new product is being designed.

This thesis was written during spring and early summer of 2021. Due to some material being confidential and intellectual property of ABB, the public version of this document will not contain such material.

2 GENERAL IDEAS OF EMBEDDED DEVICE SELF-SUPERVISION

The general purpose of self-supervision in an embedded system is to ensure that the main functionality of the device can always be executed as reliable as possible when the device is in use. In case of protection relays this means to provide disturb free distribution of electricity which includes operating the circuit breaker when needed.

Self-supervision should be designed in a way that the device is as fault tolerant as possible. This means that faults need to be categorized in some way, for example as critical and non-critical faults. When a non-critical fault is detected by self-supervision the main functionalities of the device do not need to be stopped. If the detected fault is critical, the device and possibly the whole system or process, must be able to fail in a safe way. Of course, on a process level design, a critical process should never need to be stopped in case of a single device failure, but rather have a redundant system for back-up.

Considering microcontroller-based protection relays, the things to supervise are the microcontroller itself and the volatile and non-volatile memories it uses, trip outputs and measuring functionality. With modular relays the existence of correct extension modules needs to be supervised.

Software in embedded devices, including protection relays, keeps on growing in complexity and this of course keeps increasing the possibility of something going wrong. Software needs to be robust and designed in a way that it minimizes the likelihood of failures, detect impending failures and prevent them if possible. In case of a failure software must also be able to recover from that condition.

The next sub-chapters introduce methods for supervising both the hardware and software of embedded devices.

2.1 Power-On Self Tests (POST)

POST means tests that are run when the system is powered on. In order for the device main application to start normally all of these tests need to pass. These tests naturally increase the time from power-on to the main application running but are very important since power-on process is one of the hotspots for failures to happen and often the device can process-wise fail safely during start-up.

Regarding protection relays the system start-up time is not so critical, since it is not expected to restart after it is commissioned on site, allowing POST to be comprehensive. This does not mean that POST should not be optimized to be as fast as possible and to make the system start-up as fast as possible.

The following are examples of test to be done during POST.

2.1.1 CPU, Register, Interrupt and Exception Test

Often the first test done in POST is the CPU test. This test checks the internal working of the CPU and is done by executing processor instructions and verifying the output. During this also all processor registers are tested.

Interrupt and exception processing are tested by creating conditions in which said handlers should activate. For example, a timer interrupt can be enabled that activates a flag that the test can read to verify that the interrupt handler functioned correctly. The exception handler functionality can be checked by creating exception conditions, such as divide by zero, and verifying that control has been transferred to the exception handler.

2.1.2 RAM Test

Testing the device Random Access Memory at power-on is a good practice since memory is most likely to fail at the start-up, so testing it before use is sensible. In addition, virtually nothing is yet written into the memory so the whole address space can be tested without it causing issues.

At the device start-up, it is a good practice to check memory for three type of faults:

1. **Address bus fault**
   - Not all addresses can be accessed for example due to stuck bits on the address bus. Addresses have crosstalk, meaning writing one address unwantedly affects other address or addresses.
2. **Data bus fault**
   - Not all bits on the data bus are writable and readable, due to bits stuck on either 0 or 1. Results in wrong data written to or read from an address.
3. **Data loss**
   - Data and address bus are fine, but data is lost a while later. Implicates a fault in the memory cell itself.

For all these faults a one simple test called March Test gives quite a good coverage. The test is as follows:

1. **Initializing**: Write a 0 in all memory locations on the board.
2. **Marching Ones**: Repeat the following steps starting from the lowest address until the highest address is reached:
   - Check if the content of the memory is zero
   - Write a 1 in the bit 0 position
   - Read the memory location to confirm that the bit has been written successfully
   - Repeat the above steps until a 1 has been written in all bits of that location
3. **Marching Zeros**: Repeat the following steps starting from the highest address until the lowest address is reached:
   - Check if the content of the memory is 0xFF (i.e. all bits are still set as one after the one march)
   - Write a 0 in the bit 0 position

- Read the memory location to confirm that the bit has been written successfully
- Repeat the above steps until a 0 has been written in all bits of that location /8/

2.1.3 Extension Module/Device/Card/Peripheral -Existence test

If the embedded device is modular and requires some or all modules to be attached at start-up, the existence of these modules needs to be verified during POST.

One way of doing this is by requiring the extension modules to have a form of non-volatile read-only memory in them. For example, an EEPROM component. This read-only memory is programmed by the manufacturer and contains data with what the extension module can be identified and verified by the device. During POST the device can read this content or parts of it and verify the existence of the extension module, for example, by comparing the read data or a checksum calculated from it to a pre-defined value in the device.

2.2 Runtime Tests

With most embedded devices the self-supervision keeps on monitoring the operation of the device during its whole runtime. Tests done during runtime are usually different from tests done in POST, since now the main application is running and should always have the top priority. Meaning, that self-supervision tests need to use as little processor time as possible during runtime.

What sort of tests to run depends on the main functionality of the device. With protection relays the following should be considered.

2.2.1 Trip Output Tests

A very important feature of a protection relay is the ability to operate a circuit breaker. This is usually done via the use of an output, for example in form of an

output relay. The functionality of these kinds of outputs needs to be checked periodically to ensure that when needed they can be operated.

Of course, the test itself cannot change the physical state of the output, since this could cause a false trip, which in return can cause unnecessary break in electricity distribution. Because of this the circuitry around the output needs to be designed in a way that only a feedback can be read from the output during the test without changing the actual output state.

2.2.2 Measurement Functionality Supervision

The ability to measure analog currents and voltages is another key function of a protection relay. The measurement accuracy is important, but very difficult to supervise. If the measured value is realistic but incorrect, the relay cannot really know this.

Therefore, the measurement supervision should be focused to measurements that are unrealistic. Somewhere along the line the analog values have to be converted to digital values with an ADC. The digital output values of the ADC are the ones to supervise, since these values can be unrealistic and cannot be input from the analog side.

2.2.3 Runtime RAM Test

Testing the functionality of RAM during runtime is different than during POST since now the main application is using the memory. Testing the RAM functionality is still important since failures of the memory can cause theoretically any kind of problem.

During runtime the test should reserve a fixed amount of memory to test every time the test is run. The reserved memory needs to be free, meaning the RAM test can never overrun the main application. If there is no free memory to test, the test is skipped. The test function should be simple and executed as fast as possible. For example, write a number of bytes with a fixed value, read the bytes and confirm

that the operation was successful. Keep on looping this until the whole reserved amount of memory is tested. Free the memory once test sequence is completed. /12/

2.2.4 Runtime Extension Module Existence Test

The runtime extension module existence test is similar to existence test done in POST, but this time checksum calculations should not be done in order to keep the test execution time as fast as possible. Rather it should be reading some specific data from the extension module memory and comparing this to the data that was read during POST. This way it can be verified that the extension module is still attached and functioning.

2.3 Watchdog

Supervising the device software can be a very complex process. One way to do this is by using a watchdog to make sure that the software does not spend unacceptably long time in one function, or in the worst case get stuck forever in one. A basic functionality of a watchdog is a timer which needs to be refreshed before it runs to zero, otherwise the watchdog will activate and initiate a microprocessor reset.

The watchdog should be initiated during the POST and kept on during the device runtime. In general, there are two types of watchdogs, a software watchdog and an external watchdog.

2.3.1 Software Watchdog

The software watchdog is basically a high priority function which is initiated at the device start-up and runs with the main application. It can be a timer interrupt which needs to be refreshed before it runs to zero. This way the execution time of other functions can be kept in check, and that software does not get stuck in a function. If the timer runs to zero, a microcontroller reset is initiated.

A problem with the software watchdog is that if the microcontroller completely dies, the software watchdog cannot initiate a reboot. The software watchdog is more practical when used in testing the software design and functions to make them pass required execution times.

2.3.2 External Watchdog

The external watchdog is an independent component involved in the CPU circuitry. The component is basically an external timer that needs to be refreshed by a signal coming from the microcontroller. The watchdog component has an output pin connected to the reset pin of the CPU and if the timer runs to zero, reset is initiated.

The advantage of external watchdog is that it supervises both the software and the microcontroller hardware and in case of a microcontroller fault, forces a reset on it. One example of an external watchdog is from manufacturer Maxim Integrated MAX823 5-pin component:

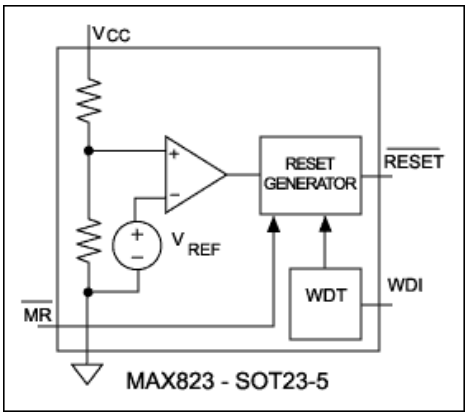

Figure 1. Maxim Integrated MAX823 external watchdog component./9/

2.4 Self-supervision Actions when Fault Detected

2.4.1 During POST

If a fault is detected during POST the first logical action would be to reset the device and try the start-up process again. Since tests done on start-up are usually on components and functions that are critical to the operation of the device, it is important for POST to pass. Of course, if the fault is permanent and POST can never pass, the device should not end up in a boot loop but try X number of times before going in to a fault-mode, where user actions are required.

Usually if the device goes through POST, it is either the commissioning of the device, manually initiated reboot or that something has gone wrong that caused the device to restart during runtime. Therefore, it is not that critical to get the device back up and running as fast as possible, but rather as reliable as possible. Naturally the faster the better.

All faults encountered during POST should be logged to a non-volatile memory on the device if possible.

2.4.2 During Runtime

When a fault is detected during runtime there should be at least two options for self-supervision actions depending on whether the fault is critical or not-so critical.

When the self-supervision detects a fault categorized as non-critical, the first action should not be something that affects the main functionality of the device, but perhaps a warning or even just a notification. If the same fault gets detected repeatedly, then some harder actions could be done. For example, a restart or even going to a “safe” mode where the device main functionality is crippled or disabled.

In case of critical faults, immediate repair actions should be taken since the correct operation of the device cannot be guaranteed anymore. An immediate restart of

the device and if same the fault appears repeatedly then going to a fault-mode is the only reasonable action left.

All faults detected during runtime should be logged to a non-volatile memory on the device.

3 COMPARISON OF PROTECTION RELAY SELF-SUPERVISION DOCUMENTATION

This part of this thesis focuses on studying documentation of self-supervision properties of protection relays from multiple manufacturers. The products, from other manufacturers than ABB, were chosen to be as similar as possible to ABB tiger series products. This study is done purely based on documentation found freely available on the manufacturers' websites without physical access or practical experience with the devices, excluding ABB relays.

3.1 ABB

The product chosen from the manufacturer ABB was a feeder protection relay REF615.

Figure 2. ABB feeder protection relay REF615. /1/

In 615 series 5.0 FP1, the following is documented about self-supervision in the technical manual:

"The protection relay's extensive self-supervision system continuously supervises the software and electronics. It handles run-time fault situation and informs the user about a fault via the LHMI and through the communication channels."/1/

Fault indications are separated to two types:

- **Internal faults (IRF)**
- **Warnings**

When IRF is detected the relay protection is disabled, the green Ready LED begins to flash and self-supervision output contact is activated. An indication message about the fault is displayed on the HMI with basic fault information, IRF code, date and time. Different actions are taken depending on the severity of fault. The first action is that the relay tries to eliminate the fault by restarting itself. If restarting does not clear the fault the relay stays in IRF mode. All other output contacts are released and locked during the internal fault. The protection relay continues to perform internal tests during the fault situation.

If an internal fault disappears, the green Ready LED stops blinking, and the relay returns to normal state.

The technical manual also includes a table of IRFs with their indications, fault codes and additional information.

In case of a warning, the relay continues to operate normally except for those functions possibly affected by the warning. The green Ready LED stays lit as during normal operation. Warnings also have an HMI indication such as IRFs. The technical manual has a similar table for warnings as for IRFs.

In document 615 series 5.0 FP1 Operation manual there is also information of relay self-supervision. It is similar to the documentation in the Technical manual with some added guidelines to troubleshooting and corrective procedures.

3.2 Arcteq

The product chosen from the manufacturer Arcteq was feeder protection relay AQ-F215.

Figure 3. Arcteq feeder protection relay AQ-F215. /5/

Documentation found about the self-supervision of this relay was very little. From the product's instruction manual following can be found:

"When the unit is powered on, the green 'Power' LED is lit. When the red 'Error' LED is lit, the relay has an internal (hardware or software) error that affects the operation of the unit."/5/

The CPU module of the relay has a system fault output relay with changeover contact with one circuit closed when there is a system fault, or the auxiliary voltage is off. It is not documented if the fault state is always permanent or can it also be temporal. The documentation does not say if the relay tries to recover from fault states or what kind of user procedure should be done when fault state is active.

3.3 Siemens

The product chosen from the manufacturer Siemens was a Reyrolle-series overcurrent protection relay 7SR51.

Figure 4. Siemens Reyrolle-series protection relay 7SR51./3/

In the 7SR51 relay device manual not that much is said about device self-supervision but following can be found /3/:

The 7SR51 has three different types of restarts:

- **Power on** = Device is switched on by applying auxiliary supply voltage.
- **Expected** = User initiated restart for example to apply a change to device configuration.
- **Unexpected** = Caused by device failure detection such as watchdog.

Here the focus will be on the unexpected restarts which are caused by failure detection. In the manual, only a watchdog is mentioned as the self-supervision function causing unexpected restarts. How this watchdog is implemented is not documented. The self-supervision of the device seems to be built around these unexpected restarts, they are counted in a parameter called **Unexpected restart count** which has a user-configurable cap, with default value being 3. The first unexpected restart starts a timer called **Unexpected restart period,** which is also user-configurable, with default value being 1 hour. If the unexpected restart count exceeds the set cap within the unexpected restart period, the device will prompt an error message and enter **locked-up** mode. In this mode all the operation of indication

LEDs, binary outputs, Human-Machine interface pushbuttons and data communications are disabled. Once locked-up mode has activated it is non-recoverable at site and device must be returned for repair.

3.4 Schneider-VAMP

The product chosen from the manufacturer Schneider Electric (VAMP) was a VAMP 321 arc-protection relay.

Figure 5. VAMP 321 arc-protection relay. /4/

VAMP 321 includes an arc self-supervision function, **ASF** for short. The purpose of ASF is to indicate to the user when the equipment may not be fully functional and user attention is demanded. ASF continuously monitors the device and cannot be disabled. The self-supervision monitors I/O units, binary inputs and outputs, FPGA configurations and auxiliary supply. The relay micro controller and the program execution are supervised by means of a separate watchdog circuit. In an inoperable situation the watchdog attempts to restart the micro controller. If the restart is unsuccessful the watchdog issues a self-supervision signal indicating a permanent internal condition. When a permanent fault is detected, all output relays are locked except for self-supervision output relay and arc protection output relays.

When ASF detects a fault, it can have two different states:

Permanent inoperative state

- If a permanent inoperative state has been detected, the device releases self-supervision relay contact and status LED is set on. The HMI will display a detected fault message. The permanent inoperative state is entered when the device is not able to handle main functions.

Temporal inoperative state

- When ASF detects an inoperative state, a self-supervision signal is sent, and an event is generated. In case the state is only temporary, an off event is generated. This state can be reset via local HMI.

No conditions are documented that when the inoperative state becomes permanent or are some types of faults instantly permanent.

4 CONCLUSIONS

The aim of this thesis was to study the self-supervision of a protection relay. This included studying general principals of embedded device self-supervision, studying documentation of self-supervision from protection relay manufacturers and finally studying and documenting the actual implementation of self-supervision in a certain protection relay.

The documentation on protection relay self-supervision was found to be similar between the manufacturers. Also, based on documentation, the effects of self-supervision noticing a fault have similar features between the studied product documentations.

The general principals of embedded device self-supervision are followed quite well in the current implementation of the studied protection relay self-supervision. Both the software and hardware are supervised at start-up and during runtime. Faults detected in critical functions cause harder actions by self-supervision than non-critical faults.

REFERENCES

/1/ 615 series Technical manual. Accessed March 2021. https://search.abb.com/library/Download.aspx?DocumentID=1MRS756887&LanguageCode=en&DocumentPartId=&Action=Launch

/2/ 615 series Operation manual. Accessed March 2021. https://search.abb.com/library/Download.aspx?DocumentID=1MRS756708&LanguageCode=en&DocumentPartId=&Action=Launch

/3/ Reyrolle 7SR51 overcurrent protection device manual. Accessed March 2021. https://cache.industry.siemens.com/dl/files/048/109773048/att_1038542/v1/7SR51_Overcurrent_Protection_Device_Manual_Version_2.20_en_US.pdf

/4/ VAMP 321 protection relay manual. Accessed March 2021. https://www.se.com/ww/en/product-range-download/62049-vamp-arc-protection/?parent-subcategory-id=4765&filter=business-6-keskij%C3%A4nnitejakelu-ja-automaatio#/documents-tab

/5/ Arcteq AQ-F215 Instruction manual. Accessed March 2021. https://www.arcteq.fi/wp-content/uploads/2019/11/AQ-F215-Instruction-manual-v2.01EN.pdf

/6/ 620 series Technical manual. Accessed March 2021. https://library.e.abb.com/public/a54dd3acc45d4cdda23b6d519b5a0bdb/620_series_tech_757644_ENf.pdf

/7/ 620 series Application manual. Accessed March 2021. https://library.e.abb.com/public/3b8e5f754bc0595bc1257b9f00173a57/REF620_appl_757651_ENa.pdf

/8/ Hardware diagnostics and Power on Self Tests. Accessed May 2021. https://www.eventhelix.com/fault-handling/hardware-diagnostics/#Device%20Tests

/9/ SUPERVISORY CIRCUITS KEEP YOUR MICROPROCESSOR UNDER CONTROL. Accessed May 2021 https://www.maximintegrated.com/en/design/technical-documents/tutorials/2/279.html

/10/ Voltage Supervisor and Reset ICs. Accessed May 2021. https://www.ti.com/lit/eb/slyy167/slyy167.pdf?ts=1621868131216

/11/ Making an embedded system safe and secure. Accessed May 2021. https://www.eetimes.com/making-an-embedded-system-safe-and-secure/

/12/ Self-testing in embedded systems: Hardware failure. Accessed May 2021. https://www.embedded.com/self-testing-in-embedded-systems-hardware-failure/