Event Detection and Disaggregation Algorithms for NIALM System

Kien Nguyen Trung[1,2], Eric Dekneuvel[2], Benjamin Nicolle[3], Olivier Zammit[3],
Cuong Nguyen Van[1], Gilles Jacquemod[2]

[1]University of Science and Technology – The University of Danang, Vietnam
ntkien@dut.udn.vn
ngvancuong2000@gmail.com

[2]Université Nice Sophia Antipolis, EPOC-UNS, Sophia Antipolis, France
{Gilles.Jacquemod, dekneuv}@polytech.unice.fr

[3]Qualisteo Company
Nice, France
{benjamin.nicolle, olivier.zammit}@qualisteo.com

Abstract—**Despite the profusion of NIALM researches and products using complex algorithms, addressing the market for low cost, compact, real-time and effective NIALM smart meters is still a challenge. This paper talks about the design of a NIALM smart meter for home appliances, with the ability to self-detect and disaggregate most home appliances. In order to satisfy the compact, real-time, low price requirements and to solve the challenge in slow transient and multi-state appliances, two algorithms are used: the CUSUM to improve the event detection and the Genetic Algorithm (GA) for appliance disaggregation. Evaluation of these algorithms has been done according to public NIALM REDD data set [6]. They are now in first stage of architecture design using Labview FPGA methodology.**

Keywords- NIALM, CUSUM, Genetic Algorithm, K-mean, classification, smart meter, FPGA.

I. Introduction

In 1992, Hart [1] first introduced Non-Intrusive Appliances Load Monitoring (NIALM) methodology that uses only one sensor to monitor all appliances in the electric network. Many problems outlined when considering intrusive method, are solved in NIALM technology, like the difficulty of installing all the system and the limit of the communication bandwidth between sensor nodes. However, NIALM still need to solve many technical challenges:

- The similarity in electrical characteristics between some appliances requires extracting more electrical signatures to distinguish them.
- Variable load appliances cause slow transients that are difficult to detect by current event detection algorithms.
- Simultaneous events are events that occur between two samplings, leading to lost events. A simple way to solve this problem is to increase the sampling frequency.
- Multi-state load appliances (for example the washing machine) have many states in their operation.

This paper is part of a research aiming at developing a new low cost, compact, real-time NIALM smart meter to address these challenges. In the next section, we will give a functional description of our proposed system and will explain technical theories underlying some critical processes. Section 3 will present many results from the functional performance evaluation REDD data set [6]. Section 4 discusses about some issues coming for efficient implementation of a prototype. Last section is about conclusions and enumerates some prospective about the future work.

II. NIALM Functional Description

A. Synchronous dataflow model of the measurement function

The Synchronous Dataflow (SDF) computational model [2] - a special case of dataflow model, is a useful model to describe Digital Signal Processing (DSP) or communication systems as a composition of nodes (sometimes called actors or processes) and arcs. Each node represents a computation and the arcs represent data and control precedence between the nodes. Fig. 1 is a dataflow model of our NIALM system with a composition of preprocessing nodes to extract power and harmonic information, event detection to detect the transient in data from the electric network, and disaggregation to classify appliances from the total apparent power.

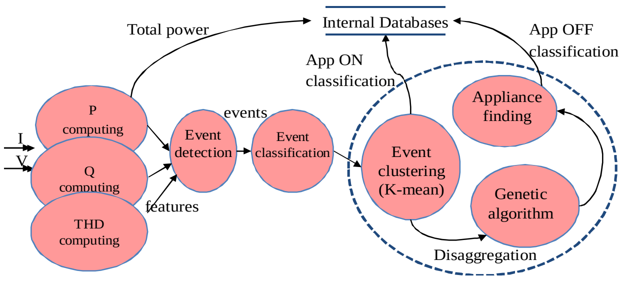

Fig. 1. Data flow model of the measurement function

Each elementary node of the graph can be implemented in hardware or in software with a suitable hardware/software co-design methodology. This method helps the system satisfy strict constraints in real-time, economically cost and low power usage. Let us now give a more detailed specification of the various processes.

B. Online Event Detection CUSUM

In 1993, Basseville and Nikiforov [3] introduced CUSUM filter to detect transients in sequence data. This algorithm is used in many applications such as monitoring petrol in tanks, tracking GPS signal, filtering noise in ear-phones, monitoring fault in DC motors etc. In fig.2, we propose an online event detection using the CUSUM adaptive filter. The Least Square low pass filter is used to predict the next input value. The predicted values are then compared to the true input values to trace out transients. The original CUSUM rule for both positive and negative transient detection is presented below:

$$g^-_k = max[0,\ g^-_{k-1} - (y_k - \hat{\theta}_{k-1}) - v]$$
$$g^+_k = max[0,\ g^+_{k-1} + (y_k - \hat{\theta}_{k-1}) - v]$$
$$t_a = min\{k : (g^+_k \geqslant h) \vee (g^-_k \geqslant h)\}$$
$$g^+_{ta} = g^-_{ta} = 0$$

Where y_k is the input value and θ_{k-1} is the estimated value of y_{k-1}. t_a is stopping time or the time where the transient occurs. Parameters *drift v* and *threshold h* are design parameters. Threshold parameter h defines the smallest change in a series of data that CUSUM can detect. Drift parameter v controls the duration from a real transient to the detected transient or the speed of the detector.

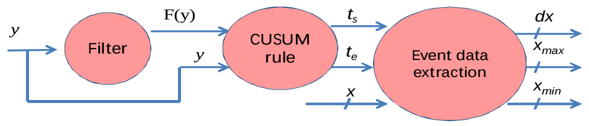

Fig. 2. Refinement of the CUSUM detection flow

We improved the CUSUM algorithm in our last research [4] to include two more features specific for NIALM system: adaptive threshold feature and detecting both the beginning and the end of the transient-state feature. Adaptive threshold h is defined by the difference between the max value and min value of the variable y during a transient. The detection of the beginning and the end of the transient is based on a simple statistic rule. The beginning of a transient happens when the stop rule occurs in steady state and the end of a transient occurs when the stop rule occurs in the transient state. The electric network is in steady state when there are not any stop rules during a predefined time while it is in transient state after a stop rule. This CUSUM detection algorithm can now detect slow transients of variable load appliances. Moreover, thanks to detailed beginning and ending of transient information, the detection can also extract any signature (x) in their change after the event (dx), their max value (x_{max}) and min value (x_{min}) during the transient and transient time ($t_e - t_s$).

C. Extracting Total Harmonic Distortion of current (THDi)

THDi is a good factor in appliance classification. However, while real power and reactive power of appliance are linear variables, which are simply the change value after the event, THDs of current depend on the phase so they are non-linear values. In real-world signal, we can represent any non-sinusoidal signal in the below format.

$$\begin{aligned} i_1(t) &= \sum_{k=1}^{\propto} I_{1k}\sin(k\omega t + \varphi_{1k}) \\ &= \sum_{k=1}^{+\propto}[Ia_{1k}\cos(k\omega t) + Ib_{1k}\sin(k\omega t)] \end{aligned} \qquad (1)$$

$$Ia_{1k} = I_{1k}\sin(\varphi_{1k}) \qquad Ib_{1k} = I_{1k}\cos(\varphi_{1k}) \qquad (2)$$

$$I_{1k} = \sqrt{Ia_{1k}^2 + Ib_{1k}^2} \qquad \varphi_{1k} = \arctan\left(\frac{Ia_{1k}}{Ib_{1k}}\right) \qquad (3)$$

$$\begin{aligned} Ia_{1k} &= \frac{2}{T}\int_0^T i_1(t)\cos(k\omega t)dt = \frac{2}{N}\sum_{t=1}^{N} i_1(t)\cos\left(\frac{2\pi}{N}kt\right) \\ Ib_{1k} &= \frac{2}{T}\int_0^T i_1(t)\sin(k\omega t)dt = \frac{2}{N}\sum_{t=1}^{N} i_1(t)\sin\left(\frac{2\pi}{N}kt\right) \end{aligned} \qquad (4)$$

where k = 1, 3, 5, .. are order of harmonics of current, and N is the number of samples in a cycle of signal.

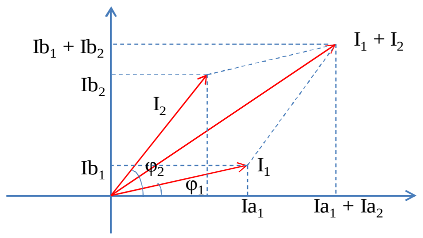

Fig. 3. Non-linear characteristic of current in fundamental part.

As illustrated in fig. 3, the total current after turning on the second appliance is defined by:

$$\begin{aligned} i(t) = i_1(t) + i_2(t) &= \sum_{k=1}^{+\propto} I_k\sin(k\omega t + \gamma_k) \\ &= \sum_{k=1}^{+\propto}[Ia_k.\cos(k\omega t) + Ib_k.\sin(k\omega t)] \\ &= \sum_{k=1}^{+\propto}[(Ia_{1k} + Ia_{2k}).\cos(k\omega t) + (Ib_{1k} + Ib_{2k}).\sin(k\omega t)] \end{aligned}$$

Linear variables Ia_{2k} and Ib_{2k} of second appliance are now the change of Ia_k and Ib_k after the detected event. We used equation (3) to estimate the amplitude and the phase of the current consumed by the detected appliance. Finally, THDi of detected appliance then is calculated by equation (5).

$$THD_I = \frac{\sqrt{\sum_{k=2}^{\infty} I_{krms}^2}}{I_{1rms}}.100\% = \frac{\sqrt{\sum_{k=2}^{\infty} I_k^2}}{I_1}.100\% \qquad (5)$$

D. Self-learning and disaggregation by Genetic Algorithm

We proposed an adaptive disaggregation method that can learn new appliances and then build a database by it-self. This disaggregation contains two algorithms: clustering transients by K-mean algorithm and matching appliances by the Genetic Algorithm (GA). At the beginning, there are no known clusters in the database. When event detection identifies a new transient, it sends event data to the disaggregation module. Because event characteristic varies by the time, the K-mean works first to compute the Euclidean distances in four dimensions coordinates of dP, dQ, dTHDi, dt between new event and detected event clusters or creates first clusters if there are not known clusters. Then, we used an adaptive threshold to define what cluster the new detected events belong to, or it is a new cluster. The detected events are stored in a temporary list. GA algorithm will process this list to find out if a group of events can form an appliance. As illustrated in fig. 4, we assume having a list of detected events data (ON/OFF) from E1 to E6. We assume that group {E2, E4, E6} is the full operation cycle of an appliance. That means this appliance has 3 statuses: E2-ON, E4-S2, E6-OFF and the summary of event information F(E2) + F(E4) + F(E6), which is called the fitness function, should be approximately equal to zero. Regular searching algorithms can solve this problem but it takes 2^6 or 64 iterations in the worst case and the long event lists consume huge computation time to do all those iterations.

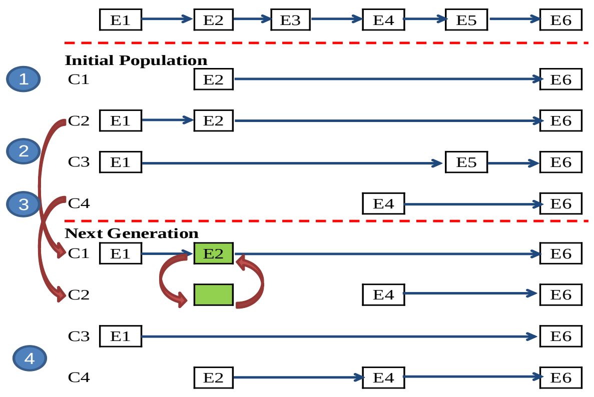

Fig. 4. Operation of GA algorithm for NIALM

GA is a random searching algorithm [5]. It processes searching with random groups to speed up the convergence of the fitness function. The algorithm includes processes below:

(1) Generate a first random population of four chromosomes.
(2) Evaluate the fitness function of each chromosome in this population.
(3) Select two chromosomes that have the best fitness.
(4) Do crossover and mutation to generate a new population from two selected chromosomes. This population is potential to contain a chromosome that better satisfies the fitness condition than the previous population.
(5) Repeat steps (2) to (4) until the fitness function is satisfied or reaching maximum iteration. In above example, it reaches the convergence only two iterations.

III. Functional Validation

We implemented event detection and tested it with the REDD public data set for energy disaggregation research of Kolter et all 2011 [6]. Fig.5 shows two cases of event detection with multi-state load appliances and variable load appliances. We then used our disaggregation to process the REDD high frequency data set in order to build the database of detected appliances as illustrated in fig. 6. We then compared it to the detected events by CUSUM in each appliance circuit of the 1Hz REDD data set in to assess how many events we missed and which appliances these missed events belong to.

Table I and table II show results of our NIALM system. The transient time information allows the distinction of electronic 1 from bathroom Ground Fault Interrupter (GFI) outlet 1 while their other characteristics are similar. THDi information improves the disaggregation between lighting and microwave oven app 2. The number of appliance in tables for example, microwave app1 and microwave app2, means that appliance has more than one operation cycles in their working. This problem is the multi-process appliances.

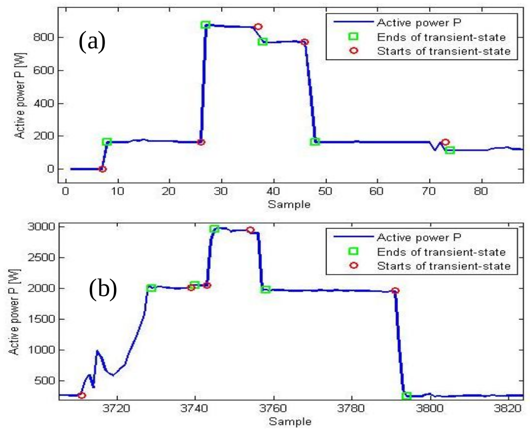

Fig. 5. (a) Detected transients of a multi-status appliance (b) Detected transients of a slow transient appliance.

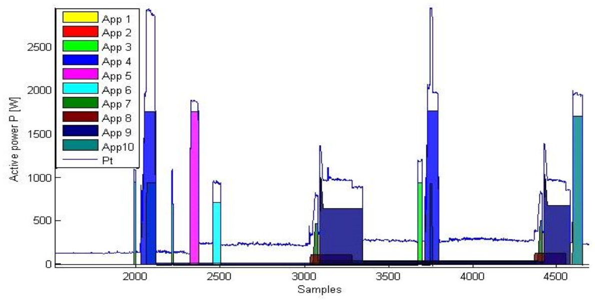

Fig. 6. Disaggregated pattern by Genetic Algorithm on REDD 16/04/2011

IV. Prototyping Through Labview FPGA

Hardware/software partitioning is a crucial step in architecture exploration. Two main constraints require a careful examination:

1. The operation duration of the preprocessing and the detection is limited by the sampling frequency.

2. The disaggregation with its three main functions should end before the next event occurs. In considering the operation of the electric network in real, it is possible to have two events occurs in about 200 milliseconds. The disaggregation then has to finish all processes in 200 milliseconds.

Labview FPGA is an environment than can be used to meet the performance constraints of a function, accelerating a normal operation in a software (CPU) processor thanks to the FPGA. Specific FPGA IC contains a huge number of elements including register, Flip-flop, BRAM, DSP cores etc which can be implemented to run in parallel to speed up the DSP process. In our application, FPGA is considered as a way to extract high-order harmonic information, or to accelerate loop processes in the NIALM system. We implemented the Preprocessing and Event Detection into both hardware and software using Labview FPGA to evaluate their performances. This performance profiling has been done considering a 400

MHz CPU and a 40 MHz Spartan 6 FPGA. The results have showed that CUSUM detection takes about 186 µs in the CPU and 1 µs in the FPGA. The first constraint requires preprocessing and detection part to run all the time and independently of other slow parts, which consume time to access the database. Relevant to the second constraint, in the worst case, GA algorithm has to run all iterations when there are no appliances in the event list. For example, if processing time of a GA cycle leads to 50 milliseconds in the worst case, considering a maximum iteration of 500, and each iteration takes about 100 µs in a 400MHz CPU. Fortunately, accelerating Kmean and GA processing using FPGA is realizable. Therefore, the system needs to run at least 3 processes in parallel as illustrated by the timeline fig.7 to meet constraints.

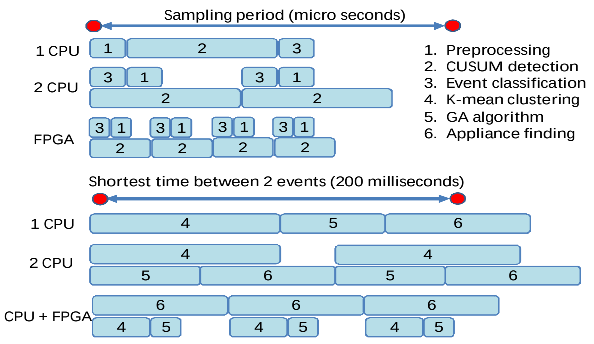

Fig. 7. Processing time estimation in some cases

From the SDF in fig.1, one architectural solution is to implement the loosely coupled Preprocessing, CUSUM Detection and Event Classification functions in a parallel pipeline of three processors competed by direct (information links) or indirect links (RAM or FIFO). We found that the minimum time to achieve these three processes is determined by the operation duration of the function 2-CUSUM detection. Then reducing the processing time of CUSUM detection is a mean to optimize performance (the throughput) by increasing the sampling frequency. Implementing result in sbRIO-9636 board allows for a sampling frequency about 5376 Hz in 400 MHz CPU and 1 MHz in 40 MHz Spartan 6 FPGA.

In overall estimation in fig. 7, all functions from 1 to 6 should be implemented in an architecture including 1 CPU and 1 FPGA to reach the highest performance. There are also embedded CPUs inside FPGA but it is necessary to study the resource usage and its impact on the price of system in comparison to a system with CPU and FPGA in separately.

TABLE I. EVENT DETECTION RESULTS IN 1-WEEK IN HOUSE 3 REDD DATA SET

Appliance type	Total event	Detected	Accuracy (%)
Furnace	205	201	98.05
Microwave Oven	80	66	82.5
Refrigerator	342	326	95.32
Washer-Dryer	136	130	95.59
Bathroom gfi	32	29	90.63
Electronic	102	100	98.04
Outlet 3	90	80	88.99

TABLE II. DISAGGREGATION RESULTS IN 1-WEEK IN HOUSE 3 REDD DATA SET

Appliance type	P [W]	Q [W]	THD [%]	dt_ON [sample]	Accuracy [%]
Washer Dryer 1	2264	29	2	11->17	95.59
Bathroom gfi 1	1693	-11	3	2->4	100
Microwave Oven 1	1690	-319	40	23->29	85.19
Outlet 3 App 3	1293	-2	3	2	76.47
Electronics App 1	1133	0	2	2	95
Bathroom GFI App 2	1065	-18	2	28	66.67
Lighting 1 App 1	1014	-120	18	25	100
Outlet 3 App 1	942	-13	3	3	87.1
Furnace App 1	666	-320	2	1	100
Furnace App 2	442	4	2	14	96.55
Outlet 3 App 2	387	2	2	10	90.91
Lighting 2 App 1	197	2	7	3	57.14
Lighting 5 App 1	180	-62	21	6	62.5
Lighting 4 App 1	132	29	51	4	72
Refrigerator App 1	123	-25	8	16->17	94.46
Microwave Oven App 2	123	-93	7	4	16.67
Furnace App 3	91	-128	12	5	93.55

CONCLUSION

In this paper, we proposed a NIALM system and explored several architectural solutions using dual processor or cooperation between processor and FPGA. Some first results show that complex algorithms like CUSUM and Genetic Algorithm extract more appliance signatures, improve NIALM to overcome some challenges such as the similarity of signatures between different appliances, slow transients in variable load appliances and multi-state appliances. These algorithms are possible to be accelerated in FPGA hardware to optimize the sampling frequency and running in real-time in a low cost system.

ACKNOWLEDGMENTS

We thank Erasmus Mundus Mobility for Asia (EMMA) organization and the company Qualisteo for their support.

REFERENCES

[1] G.W. Hart, "Nonintrusive appliance load monitoring," Proceedings of the IEEE , vol.80, no.12, pp.1870-1891, Dec 1992

[2] Lee, Edward A., and David G. Messerschmitt. "Synchronous data flow."Proceedings of the IEEE 75.9 (1987): 1235-1245.

[3] Basseville, Michèle, and Igor V. Nikiforov. Detection of abrupt changes: theory and application. Vol. 104. Englewood Cliffs: Prentice Hall, 1993.Bandyopadhyay, Sanghamitra, and Sankar Kumar Pal. Classification and learning using genetic algorithms: applications in bioinformatics and web intelligence. Springer, 2007.

[4] Trung, Kien Nguyen; Dekneuvel, Eric; Nicolle, Benjamin; Zammit, Olivier; Van, Cuong Nguyen; Jacquemod, Gilles, "Using FPGA for real time power monitoring in a NIALM system," Industrial Electronics (ISIE), 2013 IEEE International Symposium on , vol., no., pp.1,6, 28-31 May 2013

[5] Bandyopadhyay, Sanghamitra, and Sankar Kumar Pal. Classification and learning using genetic algorithms: applications in bioinformatics and web intelligence. Springer, 2007.

[6] Kolter, J. Zico, and Matthew J. Johnson. "REDD: A public data set for energy disaggregation research." proceedings of the SustKDD workshop on Data Mining Applications in Sustainability, 2011.