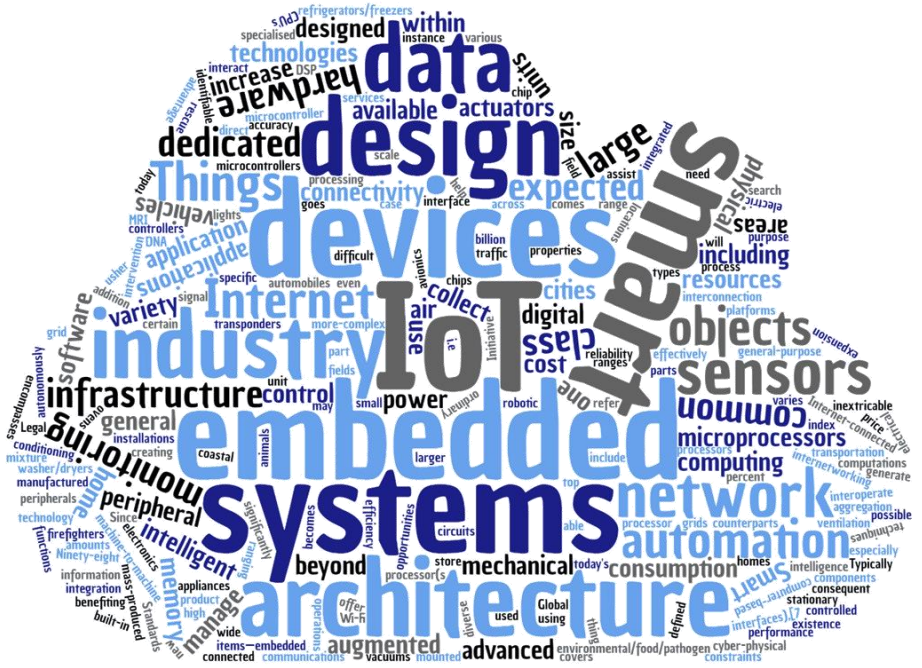
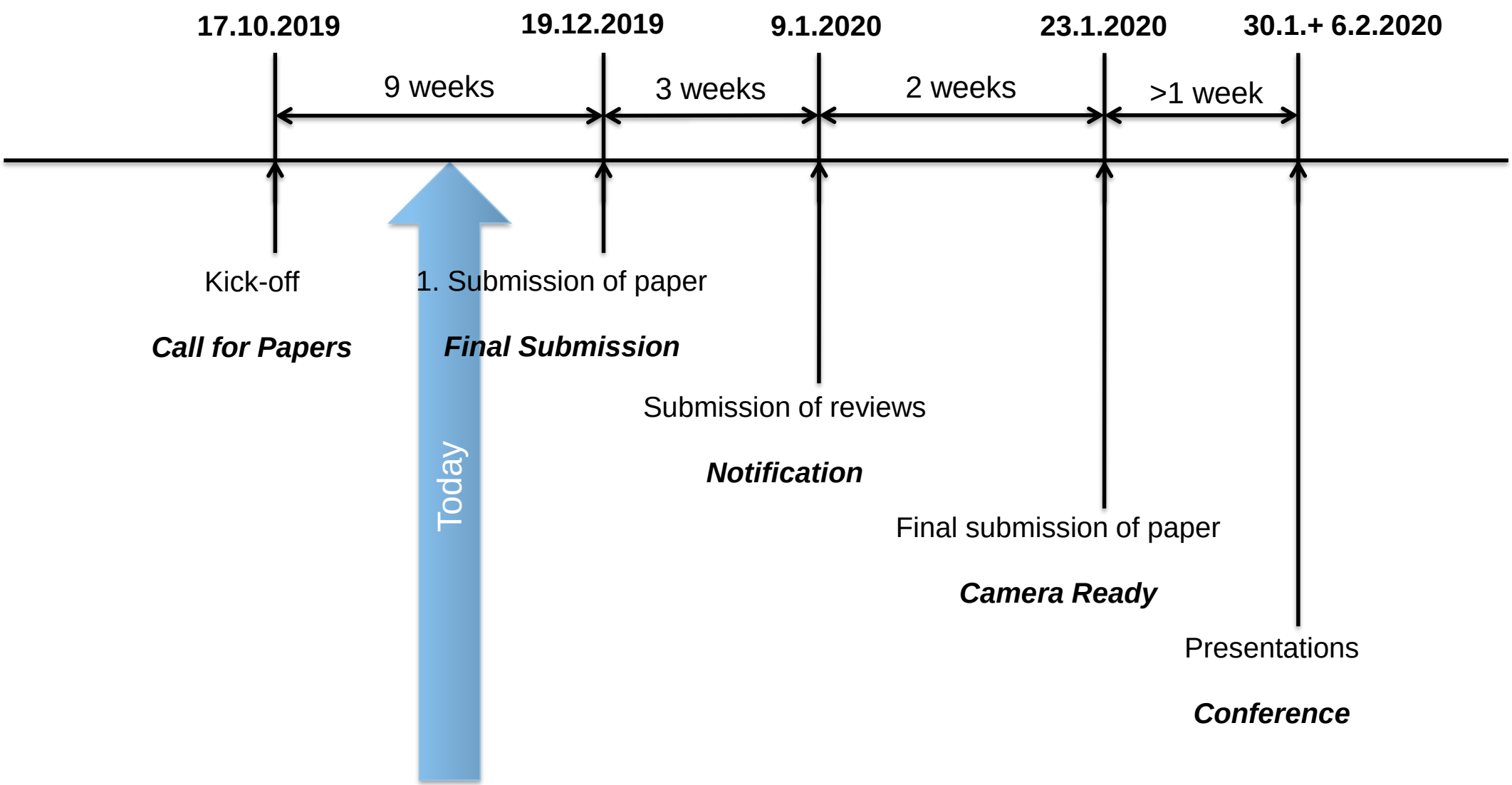
Assistant Professorship of Embedded Systems and Internet of Things
Department of Electrical and Computer Engineering
Technical University of Munich

Advanced Seminar
Embedded Systems and Internet of Things

Paper Abstract



Schedule


17.10.2019
19.12.2019
9.1.2020
23.1.2020
30.1.+ 6.2.2020
9 weeks
3 weeks
2 weeks
>1 week
Kick-off
Call for Papers
1. Submission of paper
Final Submission
Submission of reviews
Notification
Final submission of paper
Camera Ready
Presentations
Conference
Today

What makes a good Paper Abstract?

Checklist: Parts of an Abstract

Despite the fact that an abstract is quite brief, it must do almost as much work as the multi-page paper that follows it. In a computer architecture paper, this means that it should in most cases include the following sections. Each section is typically a single sentence, although there is room for creativity. In particular, the parts may be merged or spread among a set of sentences. Use the following as a checklist for your next abstract:

Motivation:

Why do we care about the problem and the results? If the problem isn't obviously "interesting" it might be better to put motivation first; but if your work is incremental progress on a problem that is widely recognized as important, then it is probably better to put the problem statement first to indicate which piece of the larger problem you are breaking off to work on. This section should include the importance of your work, the difficulty of the area, and the impact it might have if successful.

https://users.ece.cmu.edu/~koopman/essays/abstract.html

Abtract #2

Problem statement:

What *problem* are you trying to solve? What is the *scope* of your work (a generalized approach, or for a specific situation)? Be careful not to use too much jargon. In some cases it is appropriate to put the problem statement before the motivation, but usually this only works if most readers already understand why the problem is important.

Approach:

How did you go about solving or making progress on the problem? Did you use simulation, analytic models, prototype construction, or analysis of field data for an actual product? What was the *extent* of your work (did you look at one application program or a hundred programs in twenty different programming languages?) What important *variables* did you control, ignore, or measure?

https://users.ece.cmu.edu/~koopman/essays/abstract.html

Abstract #3

Results:

What's the answer? Specifically, most good computer architecture papers conclude that something is so many percent faster, cheaper, smaller, or otherwise better than something else. Put the result there, in numbers. Avoid vague, hand-waving results such as "very", "small", or "significant." If you must be vague, you are only given license to do so when you can talk about orders-of-magnitude improvement. There is a tension here in that you should not provide numbers that can be easily misinterpreted, but on the other hand you don't have room for all the caveats.

Conclusions:

What are the implications of your answer? Is it going to change the world (unlikely), be a significant "win", be a nice hack, or simply serve as a road sign indicating that this path is a waste of time (all of the previous results are useful). Are your results *general*, potentially generalizable, or specific to a particular case?

https://users.ece.cmu.edu/~koopman/essays/abstract.html

Abstract: Summary

A good abstract usually contains the following aspects (usually 1-2 sentences)

- Motivation
- Problem Statement
- Approach
- Results
- Conclusions

Discussion on examples

In this contribution an advanced methodology for model checking of analog systems is introduced. A new Analog Specification Language (ASL) for efficient property specifications is defined and model checking algorithms for implementing this language are presented. This allows verification of complex static and dynamic circuit properties like Oscillation and Startup Time that have not yet been formally verifiable with previous approaches. The new verification methodology is applied to example circuits and experimental results are discussed and compared to conventional circuit simulation.

***Abstract*—Active balancing architectures effectively increase the efficiency of large battery packs by equalizing charge between cells. For this purpose, a balancing circuit and appropriate control scheme have to be designed to enable the charge transfer via energy storage elements such as inductors. Using a manual approach to design balancing architectures can be tedious and error-prone, resulting in potentially suboptimal solutions. As a remedy, this paper presents an automatic synthesis of balancing circuits and their corresponding control, optimizing the number of required MOSFETs and necessary control signals. The proposed synthesis combines a SAT solver to explore the search space with a graph-based verification that iteratively excludes infeasible solutions until the optimal architectures are obtained. The experimental results are carried out for three given template circuits and two signal templates. The synthesis results in architectures that are superior in terms of all design objectives in comparison to solutions from literature that result from a manual design approach.**