Yes We Kanban! Introducing an Agile Methodology to Manage Your Team

Bryan Morris, P.Eng., Verilab, bryan.morris@verilab.com

Abstract—This paper describes how a management technique known as Kanban can be adopted and adapted into an ASIC/FPGA development flow. The key advantages of following such a process is that it provides a visual overview of what your team is working on, current blocking issues and bottlenecks in your process, creates opportunities for the team to proactively discuss their process and blocking issues on a regular basis, and is easy to adapt to most existing team management structures.

Index Terms—agile, kanban, lean manufacturing, ASIC/FPGA management

I. INTRODUCTION

A key verification challenge is the ability to coordinate and focus your team’s activities. Along with the increasing complexity of the ASICs and FPGAs under test, it is now common for design verification to be the critical path for many ASIC/FPGA projects. While improved tools and standardized methodologies will continue to help, I believe that it is also important to investigate other ways of organizing your team. Software teams have been successfully applying agile techniques such as eXtreme Programming (XP), Evolutionary Prototyping, Scrum and others for many years. The techniques from these methodologies have been successfully adapted in several ASIC/FPGA development flows. A more recent methodology called Kanban offers some interesting capabilities that can augment your teams existing management flow.

Kanban originated in auto manufacturing [5] that became part of the Lean Manufacturing focus. It has since been adopted (and adapted) successfully for software development. This paper provides background material to understand the value that adopting a Kanban approach to your team management can bring. It also provides guidance on adopting and adapting Kanban into your existing flow.

The key reasons to consider adopting a Kanban approach in your IC development are as follows:

• Kanban provides a visual representation of your process flow enabling your team to easily pick out where there are opportunities to optimize the work flow (i.e., in the Lean manufacturing terminology to eliminate waste).
• It highlights issues and bottlenecks in the process so that the team can apply the correct resources to resolve any issues as quickly as possible.
• It creates opportunities for discussion to ensure that the team is always working on the feature that has the highest customer value.
• It provides a more realistic view of the true current state of the project, and a view to planned completion through improved objective project metrics.

Most importantly, it is easy to adopt since you can start by simply capturing your existing process, and then incrementally optimize your team’s or project’s processes. In other words, you don’t have to change how your team is managed, but can instead gradually introduce Kanban into your flow in an incremental fashion.

If you want to know exactly what the current state your project in an easy to understand way, identify bottlenecks and quickly identify risks that become apparent during development, try using Kanban.

Lastly, while there are many tools that assist teams in following a Kanban approach, a key advantage of Kanban is that it requires nothing fancier than a whiteboard and sticky notes. This paper encourages teams to adopt this simplistic approach to start.

While web-based Kanban solutions exist, I strongly discourage their use while you are introducing this change to your team. Ensure that your team is comfortable with the techniques and underlying principles using the simple, physical whiteboard and sticky-note method before attempting an electronic approach.

II. CURRENT STATE OF AFFAIRS

Before we delve into why you should consider using Kanban, it would helpful to capture the current issues that we face in the design and verification of complex ASICs and FPGAS. These issues include, but are not limited to the following:

• increasing complexity of ASIC/FPGA development
• increasing size of project team which in turn, increases the complexity of internal communications within the team
• multi-site teams again, increasing the complexity of both team dynamics and internal communications
• understanding your project’s true current state, and
• limited opportunities to explore new techniques and methodologies while still in development.

Every project has experienced at least one of these factors. Our project management teams are highly capable individuals usually with a great deal of experience and yet most teams are incapable of producing a schedule and budget that doesn’t go off the rails shortly after starting. I know because I’ve managed many projects over many years and have spent endless hours updating the schedule to account for something unexpected, or some technical snag that’s preventing progress.

It’s time we admit that we simply cannot create a schedule that is believable. It’s not because we aren’t capable, the task is intractable because by their very nature the ASICs/FPGAs we create have too many unknowns and risks:
- **technical risks** - we are usually building something that has never been built before. Therefore, we have no prior experience to deterministically gauge its complexity.
- **technology risks** - frequently we are forced to use a new tool, methodology or process technology. In most cases, being the **bleeding edge** of technology means that we’ll need to deal with unforeseen bugs.
- **people risks** - the team we start with in most cases is not the team we end up with. Frequently, team members are shifted on and off the project, or need to support past work, or look at future possibilities.
- **resource risk** - in many large companies any physical resources such as compute farms, licenses, etc. must be shared across multiple projects. Resource contention is inevitable at some point in the project which will likely lead to unplanned for delays.

Kanban is not a silver bullet that solves all these problems. But it can help your team work in a focused, disciplined way, understand their next highest value task that must be done, and a clear understanding of what’s left to do.

III. **INTRODUCING KANBAN FOR ASIC/FGPA DEVELOPMENT**

Compared to some traditional, and perhaps, other agile project management frameworks, Kanban is extremely simple to understand. I believe that that simplicity is its core strength and why I believe using Kanban can be a force for substantive change in the way we work.

Kanban has six core practices:
1) **Visualize your Workflow**
2) **Limit Work in Progress**
3) **Manage Flow**
4) **Make Policies Explicit**
5) **Create opportunities for feedback**
6) **Improve Collaboratively, Evolve Experimentally**

Each of these are discussed in greater details in the sections below.

A. **Visualize your Workflow**

At the heart of Kanban is the idea that the team needs to see their workflow. Only by using a visual representation of your workflow can you find opportunities to improve it. In fact, Kanban is roughly translated from Japanese as **signboard** (or **visual board**).

A simple whiteboard consists of columns representing different phases in your workflow and any task or activities flow from the left (the start of your workflow) of the board to their completion on the right (when they are done). A simple example is shown in Fig. [1] where your team has defined three simple phases: **Todo**, **In Progress**, and **Done**. Similar to many Kanban boards in use, this diagram is representative of a white board with sticky-notes holding the task/activities that need to be accomplished.

Movement flows from left to right as room becomes available from right to left. As shown in Fig. [2] where **Feature 4** moves from **In Progress** to **Done**.

Leaving space in **In Progress** for **Feature 2** to be pulled in from **Todo** as shown in Fig. [3].

All activities are planned, tracked and visually represented on a Kanban board. Regular and continuous opportunities are provided to discuss as a team how to ensure that the team is focused and working on the most important activity. As shown
in Fig. 3 expedited activities such as bug fixes can be visually represented using a different colour.

A more realistic example is shown in Fig. 4. This board has multiple workflow phases:

- **Backlog**: all the features/activities that need to be done.
- **Analysis**: allowing time to understand and analyze the problem, capturing the requirements, defining the “exit” or “done” criteria, and considering different architectural approaches.
- **Design**: capturing the initial design.
- **Implementing**: implementing the solution.
- **Verifying**: debugging and/or verifying the implementation is correct and meets the criteria defined during the Analysis phase.
- **Done**: The task/activity has met the exit criteria.

![Fig. 4: More Realistic Kanban Board](image)

This board contains a separate **URGENT** workflow allowing those inevitable items that must be expedited. Of course, this “lane” should be used infrequently. This board also indicates documentation activities as a blue coloured sticky-note.

The visual aspect of a Kanban board gives rise to team discussions on how to optimize the teams processes and methodologies to eliminate wasted time in the team’s workflow.

As Kanban visualization allows the entire team to see the complete process, this is a first step to allowing the team to optimize the process. As the team works on the project, there is always time to make suggestions for adding or removing steps from the process – with the Kanban board evolving as the process evolves. Having the board be highly visible creates both a mechanism and an opportunity to discuss the process. Typically this is done during short, focused daily meetings to discuss progress and blocking issues (and ideally moving tasks through the workflow).

**B. Limit Work In Progress**

A typical approach to keeping your team busy is to give them several features to implement concurrently, while also ensuring that there is a constant flow of meetings, bugs and ‘quick’ tasks that need to be done. Clearly, this is not efficient due to the need to context switch between tasks. It is more effective to ensure that each person is focused on a smaller number of tasks and allow them to work through to completion. That is the goal of this practice to **Limit Work in Progress**.

Every project has constraints: people, compute power, licenses, planned vacation, etc. These constraints need to be clearly identified and captured on the Kanban board. This is done by assigning a work in progress (WIP) limit to each column in a Kanban board reflecting the constraints. For example, let’s assume we want to assign a WIP limit to an Implementing phase. If we have four developers and make the assumption that a developer can at most work on no more than two tasks at a time, the resulting WIP limit is 8 (= 4 developers * 2 tasks/developer). This defines the maximum number of tasks/activities that can happen during that phase of the process.

In the example Kanban board in Fig. 5 below the Implementing phase has a WIP Limit of 4, meaning that only four tasks are allowed to be in process in the Implementing phase at any one time. As shown in the figure below, the Implementing phase has reached its WIP limit at this point in the project.

![Fig. 5: Kanban board with Work In Progress Limits](image)

As can be seen by this system the addition of numeric values beside the column header indicate the WIP limits e.g., Analysis, Design and Verifying have a limit of 2 while Implementing has a limit of 4. Just to clarify the team will decide on what is an appropriate number for these columns. The initial WIP limits will start with a good estimate of what they should be, and then these numbers will likely change as the project progresses. Consider that in tweaking these values you are attempting to create a steady flow of work traversing from the start (on the left) to the finish (on the right). WIP that are too low will create a bottleneck; WIP that are too high will mean your team is working on too many things at once.

A key point here is that Kanban uses what is called a pull system. If you consider that WIP limits define the number of free slots available per column, then you can only move a task
into the column when there is a free slot. For example, in Fig. 5, above the following facts can be asserted (looking from left to right):

- The team cannot start a new item from the Backlog because the Analysis phase has hit WIP limits.
- However, once an item is complete in the Analysis phase it can immediately move into the Design phase because it has one free slot.
- Similarly, any complete item in the Design phase cannot move into the Implementing phase because it has hit its WIP limit (Note: you must include everything in the column – even the expedited lane if you have one).
- Similarly, the Verifying column cannot accept any work from the Implementing column.
- There are not limits to the Done column.

The pull concept is that tasks can only be pulled from the right; rather than being pushed from the left. As well, it is important for the team to define entry and exit criteria into each phase to ensure that you are accepting work that is complete. Accepting anything less than complete will inevitably come back to haunt you later in the project when “almost” complete work must be re-inserted back into the flow – usually at an inopportune time in the project.

An important point regarding when things move from the Backlog into the Analysis phase is that it provides an opportunity to decide which task or feature provides the most value at this time. As priorities often change throughout a project due to new information or technical issues arising, pushing the decision on what to implement next to as late as possible is key value of using a continuous system like Kanban.

Once you have WIP limits put in place, it becomes self-evident where the bottlenecks exist in your process because that particular element creates back-pressure for the up-stream activities. As discussed earlier, increasing the WIP limits will likely mean that tasks linger in a column for too long (and your team is being stretched too thinly).

Once you have an initial set of WIP limits defined, they will likely need to be tweaked as the project progresses. That is natural and should be expected. One approach is simply watch how smoothly the tasks flow across the Kanban board and experimentally increase and decrease the WIP limits to smooth that flow.

C. Manage Flow

The goal for any project is to ensure that you optimize your team as effectively as possible. The visual aspect and the WIP limits discussed in the previous sections help you in identifying bottlenecks and helping the team to optimize the flow of activities from start to finish. Kanban provides two key metrics to objectively measure your team’s productivity: lead time and cycle time. It is easiest to describe these terms visually.

The lead time is the time from when the task enters the starting phase to when it exits the completion phase i.e., as in our examples above, the time delta from when it is entered into the Backlog column to when it is moved into the Done column.

Fig. 6 illustrates the concept of a lead time.

Lead time should also be managed to ensure that you are delivering features in a timely manner. A team can optimize their lead time through any of the following ways:

- breaking a step into multiple smaller steps;
- pulling together several smaller steps into one;
- automation through scripting;
- adding additional resources e.g., compute infrastructure; and
- additional testing / review cycles to limit downstream waste activities.

A key challenge in ASIC/FPGA development is that the same feature can be more complex in one domain than another. For example, a particular feature can be harder to implement in RTL than it is to verify, and vice-versa. Lead time can be managed on ASIC/FPGA features by breaking the task into applicable RTL, verification, etc. domains for your project. However, a trailing integration task should be included to ensure that when you say a feature is done, it includes the integration of the separate domains into a complete feature.

Another metric is cycle time that measures when work actually started on the task, and when it is completed. This is not measuring effort, but duration that the cycle stayed in the system. Fig. 7 overlays the cycle time on the lead time definition.

From the figure, it is clear that cycle time is the time it takes for a feature to move from beginning to end of your process, while lead time is the time between when the feature enters and exits your process. Calculating the average lead time and cycle time provides insights into your work flow. Average lead time can provide empirical proof that your process is improving as it gives a sense of the speed of delivery. While cycle time can provide objective measurement of how long it takes to deliver
a feature which can be used in future feature estimation and planning.\(^1\)

Once you have gathered a good set of empirical data with your lead times and cycle time you can then apply Little’s Law (from queueing theory) to calculate your team’s throughput.

**Little’s Law** The long-term average number of customers in a stable system \(L\) is equal to the long-term average effective arrival rate \(\lambda\), multiplied by the average time a customer spends in the system, \(W\), or expressed algebraically: \(L = \lambda W\)

Or expressed in terms for Kanban:

\[
AvgCycleTime = \frac{Avg.WorkinProgress}{Avg.Throughput}
\]

where **Throughput** is the average departure rate from the board.

As you tweak your process your throughput should reflect whether the change improved your work flow or not.

Using the last three core practices, the team can now visually see the following problems in their workflow:

- identify the bottlenecks
- when a task stays in a particular phase (or column) of the workflow for longer than normal
- where a process flow can be optimized

The rule to **manage your flow** is intended to focus your team on the fact that your **entire** process should be continuously reviewed, looking for ways where your team can be made more effective and productive. In other words, the key is to find and eliminate **wasteful** activities in your process.

For ASIC/FPGA teams it is encouraged that before you start the Kanban process on your next project that everyone has a clear understanding of the **big picture** for the project. This should include the set of prioritized features that are expected to be delivered, and an initial architectural view of how all the features work together. To be clear this architectural view should be all the pieces of the system, including but not limited to RTL design, verification, physical design, and software, if applicable.

By getting this view, and keeping it aligned with the **actual** architecture as it evolves throughout the project, the team can then provide some guidance on how the co-dependent features should be developed. This reduces the likelihood of rework due to building something incorrectly.

1) **Examples of Waste**: A key element of Kanban and **Lean Manufacturing** was to put an emphasis on eliminating waste. In a design environment this translates into wasteful activities such as:

- **doing more than needed**: “Hey, this feature will be handy someday, let’s add it in.”
  
  The team should only focus on what’s needed now.
  
  Completing the task in the most optimal way. This does not mean to **preclude** the very valuable task of architecting a module/class to accommodate future features, but to recognize that only work that adds value should be considered.

- **make work**: only create what’s absolutely needed to complete the work. Doing work that doesn’t contribute to the task at hand is a waste of effort.
  
  We all know that ASICs are expensive projects. Therefore, any extra or useless work should be **strongly** discouraged by the entire team.

- **multi-tasking**: cost of the context switch.
  
  Kanban’s WIP limits provides some limits to stretching your team too thinly.

- **almost-done**: not being completely done (“done, done”).
  
  Kanban fosters the idea of clearly defining when a task is done with entry and exit criteria from every column.

- **waiting for someone**: “I can’t start that task until I get it from Alice.”
  
  A visual board and daily meetings creates an opportunity to indicate blocking tasks. Or to put a more positive spin on it, the opportunity for your team to discuss ways of avoiding blocking tasks by seeking solutions as a team.

  A typical example in an ASIC project is when design is waiting for a verification environment. Kanban can help with this by splitting the verification tasks into creating a simple “sanity” verification environment that is **good enough** for designer’s needs, and then adding tasks that adds more complexity to a verification environment.

- **wrong resource applied to the wrong task**: “Get Bob to do this, he’s not busy.”
  
  Assigning someone not capable of doing the job because they are available usually ends in sub-par quality work that takes longer than desired, and inevitably needs to be reworked later.\(^2\) Any rework is a waste. While Kanban does not provide any mechanism to highlight this issue, it may assist in flagging this as an issue since the task will likely take longer than usual.

**D. Make Policies Explicit**

The goal of this core practice is to clearly articulate any underlying policies in your work flow i.e, how things are really done. Once articulated these become the ‘facts’ for the project, and therefore reducing any subjective analysis or emotional baggage attached to that step in the process. As the saying goes:

The best disinfectant is sunshine.

meaning in this case\(^3\) that only once polices are in the open and explicit can you define why they are problematic.

These explicit policies can be discussed objectively with the goal to improving the process. New process suggestions once implemented can then become the new policies. Providing clarity on how things **should** work avoids some wasteful activities.

\(^1\) Assuming that you define a “size” for each feature. Many teams use crude but effective T-shirt sizing for tasks i.e., small, medium, large and extra-large. Then you can calculate a cycle time for all medium tasks. When planning for a medium task you can then use the average cycle time for the medium task to estimate how long it will take. Using empirical data will result in better planning estimates than how many traditional teams do it.

\(^2\) Unless this is viewed as a training opportunity – which has great value.

\(^3\) Yes, it has other meanings.
E. Create opportunities for feedback

It is important to create opportunities to review your process with objective evidence gathered, or anecdotal based on what actually happened. These opportunities can be used to create ideas for improvement that everyone can agree upon – because you’ve focused the discussion on the facts. Ideally, this feedback is done as soon as you have gathered sufficient evidence to have a meaningful discussion – which could be measured in weeks or months.

When bottlenecks occur it is also an opportunity to discuss ideas. These ideas can then be put into your backlog (or todo) list to be prioritized and added when the next available slot in your first phase is free.

As well, the daily meeting provides an opportunity to discuss issues in an objective way. The Kanban board being a physical element allows people to talk about the “board” rather than about a “person”. By transferring the discussion about moving a task across the board (“What’s preventing task A from moving forward?”), it eliminates some of the people issues around (“Why are you taking such a long time with task A?”). Discussion around those kind of issues are better suited to a different forum than a team meeting.

F. Improve Collaboratively, Evolve Experimentally

Kanban provides some key metrics such as cumulative flow diagrams (CFD) that can provide a realistic view on the current state of the project, some insight into the projected schedule going forward, and some key metrics to assist when doing forward planning. All of this information dovetails nicely into existing management reporting structures and can provide a more realistic view of true project status than a Gantt chart approach.

A cumulative flow diagram is calculated on a daily basis by summing the total number of tasks in each column, plotting that data onto a graph on a daily basis. The resulting diagram (see Fig. 8 illustrating each phase’s daily task sum in a different colour).

The following figure is an excellent (open-source) view of the information that can be extracted from a CFD.

It may be difficult to see at this size, but the lead time and cycle times can be extracted from this diagram as horizontal lines:

- lead time is the distance from the left-most of the first phase (Requirements and the left-most point on the last phase (Test).
- cycle time is the distance from the right-most of the first phase and the left-most point of the last phase.

Similarly, the amount of work left to complete can be seen as the difference vertically between the Test and Requirements. In fact, by plotting the trend lines for the first and last phases you can get an estimate of how much effort is left before you complete the current set of features from the point where the two trend-lines intersect. Of course, this is a very coarse scheduling methodology due to the fact

Fig. 8: Cumulative Flow Diagram

that it does not take into account the complexity and size of each feature or task in the backlog, nor of any future issues that inevitably arise.

As Kanban is a continuous flow, you can at any point have a discussion about changing an element in the process work flow. Typically, this is due to a chronic condition of one phase in the process being a bottleneck – requiring some consideration from the team on how to eliminate (or ameliorate) the bottleneck. Ideally, this done from a position of strong empirical data showing a persistent bottleneck so that rational fact-based decisions can be reached – such as the CFD.

There are other metrics that can be gathered, but are beyond the scope of this article.

Once the idea for improvement has been decided upon it should be scheduled like any other task by adding into your initial phase e.g., Backlog, and then implemented. As long as you continue to gather metrics your team will be able to objectively determine if the idea improved or degraded your team’s performance.

The underlying theory behind this core practice is that you identify issues taking a long-term view, make changes based on facts, and can then see whether the changes actually improved your team’s productivity.

IV. INTEGRATING KANBAN

This section provides guidance on how a team can integrate the Kanban approach into their current workflow. This is based partially on my experience and other case studies from the software development community.

Kanban compared to other Agile methods is a very low overhead and simple methodology with simple rules that can be easily adopted. The fact that it does not prescribe the steps you’ll take in your process is part of its attraction because it can be added into your existing work flow and then the team can self-organize and, if necessary, continue to improve that flow.

While this article has only scratched the surface to Kanban, it also should be sufficient for a team to start to use it
effectively. Advanced information can be found from some of the material provided in the bibliography.

The remainder of this section attempts to identify what can be done to incrementally adopt Kanban into your flow.

There are three Japanese terms to describe the movement from novice to master:

• **shu**: where we learn the basics of a technique or methodology,

• **ha**: where we understand the basics and incrementally adopt and adapt new techniques, and

• **ri**: where we have mastered the technique, use advanced metrics and can innovate.

The closest equivalent Western phrase is probably *crawl, walk, run*.

### A. Shu - Introducing Kanban

The suggested phases to introduce Kanban to the team are as follows:

- **Phase 1 - Training**: Find a champion or champions to learn about Kanban. An excellent grounding can be found in [2].

- **Phase 2 - Visualize your Flow**: Start with writing down your current process using a whiteboard. That is, start with how your team currently work *right now*. Not how you **want** to work, but how you **actually** work. This is a good place to start. Ensure that the team understands the entry and exit criteria for each phase.

- **Phase 3 - Add your WIP Limits**: Start adding WIP limits – grounded in reality. Good starting examples are to start with the number of engineers in your group and then how many tasks of each type they could reasonably handle. It is a good idea to start low and work up when the evidence indicates increasing the WIP is a good idea e.g., when free slots remain open for too long, or the cycle time is too long. This starts to identify the bottlenecks and resource constraints.

- **Phase 4 - Review**: This can be done to varying degrees throughout all the stages. The intent here is to encourage discussions around the process and the bottlenecks. The goal is to consider *incremental* improvements to whatever it is you are doing.

### B. Ha - Using Kanban

Once you acquired basic skills in Kanban you can begin to adopt

- **Capture Metrics**: Start capturing the metrics identified in this article and in the reference material. This can provide objective data from which pragmatic changes can be applied. As well, it provides a baseline to objectively determine if any process change have actually improved your team’s productivity.

- **Use Metrics**: Use the metrics collected during any discussions (or create opportunities to discuss the metrics) so that you use empirical evidence to improve your estimation of tasks/activities and for planning how much work your team can reasonably handle.

### C. Ri - Mastering Kanban

At this point, your team is fully versed in using Kanban. Using your collected data you can now incorporate these into your future planning sessions to estimate the duration of the set of tasks before you. Your planning is now driven from empirical evidence rather than hope-filled guesses. Since the estimates are a based on empirical evidence you will have greater confidence in their accuracy.

There are other techniques that are outside the scope of this paper.

### D. Kanban Caveat

Change is hard, disruptive and can be uncomfortable for many different types of people on your team. It is strongly advised that you adapt Kanban (or any new process) in an incremental way. Be aware of the significant people issues around introducing a new technique. Most people will react poorly when the change is not fully understood. By training and introducing Kanban gradually, you will have a greater chance of success.

### V. Conclusion

In summary, Kanban is a simple process tool that can provide ASIC/FGPA teams with flexibility and power to manage their flow of work. The tools discussed in this paper are the starting points to introducing Kanban incrementally, and then as team working on optimizing your development process using solid metrics to provide insights into the reasons behind the change, and validation that the change was beneficial.

Finally, discussion on Kanban in this article is meant only to serve as an introduction. While Kanban itself is relatively straightforward, the motivations for using it and the context in which it was developed may require more explanation. For more on Kanban and how Kanban is applied in lean manufacturing and product development, refer to the further reading such as [1], [2], [3], [4].

### References


