# Thesis Title

## Candidate Full Name

Thesis to obtain the Master of Science Degree in

## Aerospace Engineering

| | |
|---|---|
| Supervisor(s): | Prof. Full Name 1 |
| | Dr. Full Name 2 |

## Examination Committee

Chairperson: Prof. Full Name
Supervisor: Prof. Full Name 1 (or 2)
Member of the Committee: Prof. Full Name 3

**Month Year**

Dedicated to someone special...

# Acknowledgments

A few words about the university, financial support, research advisor, dissertation readers, faculty or other professors, lab mates, other friends and family...

# Resumo

Inserir o resumo em Português aqui com o máximo de 250 palavras e acompanhado de 4 a 6 palavras-chave...

**Palavras-chave:** palavra-chave1, palavra-chave2,...

# Abstract

Insert your abstract here with a maximum of 250 words, followed by 4 to 6 keywords...

**Keywords:** keyword1, keyword2,...

# Contents

# List of Tables

# List of Figures

# Glossary

**TLS** Transport Layer Security. 1, 2, 3, 4

**SSL** Secure Sockets Layer.

**IETF** Internet Engineering Task Force.

**MAC** Message Authentication Code.

**PSK** Pre-Shared Key.

**RPK** Raw Public Key.

**AEAD** Authenticated Encryption With Associated Data.

**PKC** Public Key Cryptography.

**HMAC** Hash-Based Messaage Authentication Code.

**HKDF** HMAC-based Extract-and-Expand Key Derivation Function.

**HTML** Hypertext Markup Language.

**HTTPS** Hypertext Transfer Protocol Secure. 2

**ECC** Elliptic Curve Cryptography.

**IV** Initialization Vector.

**ECDH** Elliptic Curve Diffie-Hellman.

**ECDHE** Elliptic Curve Diffie-Hellman Ephemeral.

**ECDSA** Elliptic Curve Digital Signature Algorithm. 1

**RFC** Request For Comment.

**PRF** Pseudo-Random Function.

**RSA** Rivest-Shamir-Adleman. 1

**DH** Diffie-Hellman.

**PMS** premaster secret.

**DSA** Digital Signature Algorithm.

**PFS** Perfect Forward Secrecy. 4

**MITM** Man In The Middle.

**AC** Asymmetrical Cryptography.

**SC** Symmetrical Cryptography.

**IoT** Internet Of Things. 1, 2, 3

**DTLS** Datagram TLS. 1, 2, 3, 4

**(D)TLS** (D)TLS. 3

**CoAP** Constrained Application Protocol. 3

**EC** Elliptic Curve.

**SCA** Side-Channel Attack.

**OCSP** Online Certificate Status Protocol.

**CRL** Certificate Revocation List.

**CA** Certification Authority.

**SNI** Server Name Indication.

**DoS** Denial-of-Service.

**DDoS** Distributed Denial-Of-Service. 1

**PKI** Public Key Infrastructure.

**AE** Authenticated Encryption.

**NSA** US National Security Agency.

**APK** Authorized Public Key.

**IANA** Internet Assigned Numbers Authority.

**JIT** just-in-time.

**IR** Intermediate Representation.

**PAPI** Performance Apppicaiton Programming Interface.

# Chapter 1

# Introduction

## 1.1 Introduction

In recent years there has been a sharp increase in the number of IoT devices and this trend is expected to continue[1]. The IoT is a network of interconnected devices, which exchange data with one another over the internet. In fact, it can be any object that has an assigned IP address and is provided with the ability to transfer data over a network. While there are many types of IoT devices, all of them are restricted: they have limited memory, processing power and available energy. Examples of IoT devices include temperature sensors, smart light bulbs and physical activity trackers. Even a salt shaker[2] can now be part of the global network.

The IoT technology provides many benefits, from personal comfort to transforming entire industries, mainly due to increased connectivity and new sources for data analysis. The technological development, however, tends to focus on innovative design rather than on security. IoT devices frequently connect to networks using inadequate security and are hard to update when vulnerabilities are found.

This lack of security in the IoT ecosystem has been exploited by the the *Mirai* botnet[3] when it overwhelmed several high-profile targets with massive DDoS attacks. This is the most devastating attack involving IoT devices done to date. However, the *Reaper* botnet[4] could be even worse if it is ever put to malicious use. Similar attacks will inadvertently come in the future.

In the process of the work on this dissertation, we have made several contributions to the TLS $1.3$ specification, and were formally recognized as contributors[5]. Our name can be found in the document specifying TLS $1.3$[6]. Although to the lesser extent, we have also contributed to DTLS $1.3$ specification[7]. We have found a security issue within the TLS implementation of the *mbedTLS* library. We reported it and it has been assigned a *CVE* with the id *CVE-2018-1000520*[8]. It is an authentication problem, where certificates signed with an incorrect algorithm were accepted in some cases. More specifically, *ECDH(E)-RSA* ciphersuites allowed ECDSA-signed certificates, when only RSA-signed ones should have been. We also found a bug in *mbedTLS*'s test suite related to the use of deprecated *SHA-1*-signed certificates and submitted a code fix to it[9][10].

### 1.1.1 Motivation

While inter-device communication has numerous benefits, it is important to ensure the security of that communication. For example, when you log in to your online banking account, you do not want others to be able to see your password, as this may lead to the compromise of your account. Having your account compromised means that a malicious entity might steal your money. Similarly, when you are transferring funds via online banking, you want the contents of that operation to be invisible to an observer, for privacy reasons. It is also desirable that no party is able to tamper with the data en transit, as it may lead to undesired consequences, such as the transfer of a larger amount than intended. Proper communication security allows those goals to be achieved.

TLS is one of the most used protocols for communication security. It powers numerous technologies, such as HTTPS. TLS offers the security services of authentication, confidentiality, privacy, integrity, replay protection and perfect forward secrecy. It is not a requirement to use all of those services for every TLS connection. The protocol is similar to a framework, in the sense that you can enable individual security services on a per-connection basis. For example, when you are downloading software updates, while data confidentiality is probably not a concern, data authenticity and integrity, are. In TLS, it is possible for a connection to only offer authenticity and integrity, without offering confidentiality. Foregoing unnecessary services will lead to a smaller resource usage, which in turn leads to smaller execution time and power usage. This is especially important in the context of IoT, due to the constrained nature of the devices.

Existing work does not explore the computational costs of the security services available in TLS. Examples of such costs are the number of CPU cycles executed, time taken and power used. Thus, developers wishing to deploy the TLS protocol in constrained environments do not have a resource that would help them in choosing a TLS configuration appropriate to the environment's needs and limitations.

TLS is designed to run on top of a reliable, connection-oriented protocol, such as TCP. DTLS is the version of TLS that runs on top of an unreliable transport protocol, such as UDP. Most IoT devices have very limited processing power, storage and energy. Moreover, the performance of TCP is known to be inefficient in wireless networks, due to its congestion control algorithm. This situation is worsened with the use of low-power radios and lossy links found in sensor networks. Therefore, in many cases the use of TCP with IoT is not the best option. For this reason, DTLS, which runs on top of UDP, is used more frequently in such devices. While the work of this dissertation will be focused on TLS, the majority of it can also be applied to DTLS. This is a consequence of DTLS being just an adaption of TLS over unreliable transport protocols, without changes to the core protocol.

There are numerous IoT devices, each one with different hardware capabilities and security requirements. For example, some IoT devices have the resources to use public key cryptography, while for others symmetric cryptography is the only option. In some cases, the communicating devices require data authenticity, confidentiality and integrity (e.g. when logging in into a device), while in others data authenticity and integrity is enough (e.g. when transferring updates).

TLS was not designed for the constrained environment of IoT. Despite that, it is a malleable protocol and can be configured to one's needs. In essence, it is a combination of various security algorithms that together form a protocol for communication security. If configured properly, it is possible to use it in the context of IoT.

The majority of existing work on (D)TLS optimization proposes a solution that is either tied to a specific protocol, such as CoAP, or requires an introduction of a third-party entity, such as the trust anchor in the case of the S3K system[11] or even both. This has two main issues. First, a protocol-specific solution cannot be easily used in an environment where (D)TLS is not used with that protocol. Second, the requirement of a third-party introduces additional cost and complexity, which will be a big resistance factor in adopting the technology. This is especially true for developers working on personal projects or projects for small businesses, leaving the communications insecure in the worse case scenario. Therefore a solution that is protocol independent and fully compatible with the (D)TLS standard and existing infrastructure is desired.

Another issue with the existing literature is that it almost exclusively focuses on DTLS optimization and not all of it can be applied to TLS. Herein we want to further explore TLS optimization. There is clearly a need for that, especially with CoAP over TCP and TLS standard[12] being currently developed. The mentioned standard does not explore any TLS optimizations, and since any IoT device using it in the future would benefit from them, this is an important area to explore.

### 1.1.2 Objectives

(D)TLS is a complex protocol with numerous possible configurations. Each configuration provides different set security services and a different security level. This has a direct impact on the resource usage. Thus, the cost of a (D)TLS connection can be lowered, by using an appropriate configuration. Typically, this involves making security/cost trade-offs. Optimizing the connection cost by selecting one of the numerous configurations available in (D)TLS meets our goals of being protocol independent, fully compatible with existing infrastructure and targeting TLS optimization specifically.

The objective of this work is to provide a means of assisting application developers who wish to include secure communications in their applications to make security/resource usage trade-offs, according to the environment's needs and limitations. We will give a general overview of of the costs of the TLS protocol as a whole and of its individual parts. This is will allow us to answer questions such as *How much will we save if we use protocol X instead of Y for authentication?*. Thus, performing evaluations on specific IoT hardware or analyzing TLS-specific optimizations on it is outside of the scope of this paper.

In order to achieve our goal, the cost of each individual security service will be evaluated. With this information, the programer will be able to choose a configuration that meets his security requirements and device constraints. If the limitations of the device's hardware do not allow to meet the requirements, the programer may decide on an alternative configuration, possibly with a loss of some security services and a lower security level, or forgo using (D)TLS altogether. Thus, this work is targeted towards developers and InfoSec professionals who wish to add communication security to applications in the IoT environment.

In our work, we evaluated the *mbedTLS 2.7.0*'s[13] implementation of the TLS protocol version $1.2$.

*mbedTLS* is among the most popular libraries with a TLS implementation for embedded systems. TLS protocol version $1.2$ is currently the most used version of TLS on the internet [14]. The work on the dissertation started before TLS protocol's version $1.3$ specification was finished and there were no embedded device libraries which implemented it. For this reason we did not evaluate TLS $1.3$. Despite that, the results obtained in this work apply to it as well, since the core functionality of the security services remained mostly unchanged.

We used two cost metrics: the estimated number of CPU cycles and the time taken. The time values were read direcly from the processor's registers. The profiled instructions are CPU-bound, thus the number of CPU cycles will be proportional to time, as will later show in our analysis. Later in the text we will show that our esitmates do reflect real values, by comparing them to time measurments obtained directly from the CPU registers. Section **??** contains a detailed description of the evaluated costs and their limitations.

### 1.1.3 Results

In summary, the results of this work are enumerated as follows:

1. Evaluate the costs of the security services of confidentiality, integrity, PFS and authentication in TLS
2. Evaluate and compare the costs of various alternative algorithms which can be used to provide each one of the security services
3. Evaluate and compare the costs of all of the possible TLS configurations present in *mbedTLS 2.7.0*
4. Contribute to the TLS protocol's version $1.3$ specification
5. Contribute to the DTLS protocol's version $1.3$ specification
6. Find and report a security vulnerability present in *mbedTLS 2.7.0*
7. Find, report and submit a patch to fix a bug present in *mbedTLS 2.7.0*

### 1.1.4 Structure of The Document

The document is organized as follows: Section 2 describes the background. It introduces some of the concepts that will be used throughout the document. Section 3 describes the TLS and DTLS protocol versions $1.2$ and $1.3$, with a focus on the version $1.2$ since it is the latest and the most used version of the protocol at the time we started this work (version $1.3$ was still in draft mode). Section 4 describes all of the related work done in the area and the current state of the art. Section 5 describes the objectives of the work and evaluation are described in more detail. Section 6 covers the evaluation's methodology and limitations. In Section 7 we evaluate the costs of the various TLS configurations and their individual parts. Finally, the conclusion of the work is done in Section 8.

# Chapter 2

# Background

Insert your chapter material here...

## 2.1 Theoretical Overview

Some overview of the underlying theory about the topic...

## 2.2 Theoretical Model 1

The research should be supported with a comprehensive list of references. These should appear whenever necessary, in the limit, from the first to the last chapter.

A reference can be cited in any of the following ways:

- Citation mode #1 - [**?** ]
- Citation mode #2 - **?** ]
- Citation mode #3 - [**?** ]
- Citation mode #4 - **?** ]
- Citation mode #5 - [**?** ]
- Citation mode #6 - **?**
- Citation mode #7 - **?**
- Citation mode #8 - **?**
- Citation mode #9 - **?**
- Citation mode #10 - [**?** ]

Several citations can be made simultaneously as [**? ?** ].

This is often the default bibliography style adopted (numbers following the citation order), according to the options:
`\usepackage{natbib}` in file `Thesis_Preamble.tex`,
`\bibliographystyle{abbrvnat}` in file `Thesis.tex`.

Notice however that this style can be changed from numerical citation order to authors' last name with the options:
`\usepackage[numbers]{natbib}` in file `Thesis_Preamble.tex`,
`\bibliographystyle{abbrvunsrtnat}` in file `Thesis.tex`.

Multiple citations are compressed when using the `sort&compress` option when loading the `natbib` package as `\usepackage[numbers,sort&compress]{natbib}` in file `Thesis_Preamble.tex`, resulting in citations like [**? ? ? ?** ].

## 2.3 Theoretical Model 2

Other models...

# Chapter 3

# Implementation

Insert your chapter material here...

## 3.1 Numerical Model

Description of the numerical implementation of the models explained in Chapter 2...

## 3.2 Verification and Validation

Basic test cases to compare the implemented model against other numerical tools (verification) and experimental data (validation)...

# Chapter 4

# Results

Insert your chapter material here...

## 4.1 Problem Description

Description of the baseline problem...

## 4.2 Baseline Solution

Analysis of the baseline solution...

## 4.3 Enhanced Solution

Quest for the optimal solution...

### 4.3.1 Figures

Insert your section material and possibly a few figures...

Make sure all figures presented are referenced in the text!

**Images**

Figure 4.1: Caption for figure.

(a) Airbus A320

(b) Bombardier CRJ200

Figure 4.2: Some aircrafts.

Make reference to Figures 4.1 and 4.2.

By default, the supported file types are *.png,.pdf,.jpg,.mps,.jpeg,.PNG,.PDF,.JPG,.JPEG*.

See `http://mactex-wiki.tug.org/wiki/index.php/Graphics_inclusion` for adding support to other extensions.

### Drawings

Insert your subsection material and for instance a few drawings...

The schematic illustrated in Fig. 4.3 can represent some sort of algorithm.

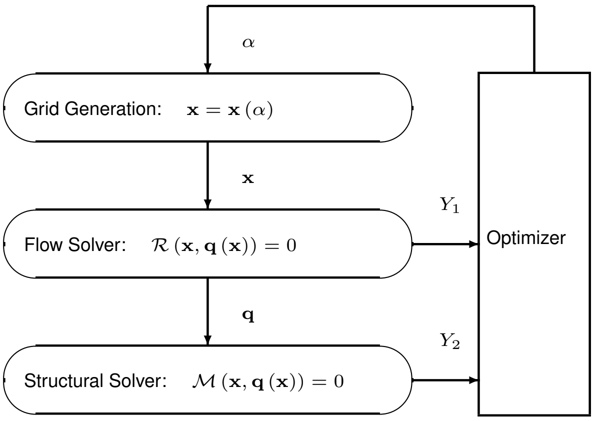

$\alpha$
Grid Generation: $\mathbf{x} = \mathbf{x}(\alpha)$
$\mathbf{x}$
$Y_1$
Flow Solver: $\mathcal{R}(\mathbf{x}, \mathbf{q}(\mathbf{x})) = 0$
Optimizer
$\mathbf{q}$
$Y_2$
Structural Solver: $\mathcal{M}(\mathbf{x}, \mathbf{q}(\mathbf{x})) = 0$

Figure 4.3: Schematic of some algorithm.

## 4.3.2 Equations

Equations can be inserted in different ways.

The simplest way is in a separate line like this

$$\frac{\mathrm{d}q_{ijk}}{\mathrm{d}t} + \mathcal{R}_{ijk}(\mathbf{q}) = 0\,. \tag{4.1}$$

If the equation is to be embedded in the text. One can do it like this $\partial\mathcal{R}/\partial\mathbf{q} = 0$.

It may also be split in different lines like this

$$\begin{aligned} \text{Minimize} \quad & Y(\alpha, \mathbf{q}(\alpha)) \\ \text{w.r.t.} \quad & \alpha\,, \\ \text{subject to} \quad & \mathcal{R}(\alpha, \mathbf{q}(\alpha)) = 0 \\ & C(\alpha, \mathbf{q}(\alpha)) = 0\,. \end{aligned} \tag{4.2}$$

It is also possible to use subequations. Equations 4.3a, 4.3b and 4.3c form the Naver–Stokes equations 4.3.

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j\right) = 0\,, \tag{4.3a}$$

$$\frac{\partial}{\partial t}\left(\rho u_i\right) + \frac{\partial}{\partial x_j}\left(\rho u_i u_j + p\delta_{ij} - \tau_{ji}\right) = 0, \quad i = 1,2,3\,, \tag{4.3b}$$

$$\frac{\partial}{\partial t}\left(\rho E\right) + \frac{\partial}{\partial x_j}\left(\rho E u_j + p u_j - u_i \tau_{ij} + q_j\right) = 0\,. \tag{4.3c}$$

### 4.3.3 Tables

Insert your subsection material and for instance a few tables...

Make sure all tables presented are referenced in the text!

Follow some guidelines when making tables:

- Avoid vertical lines
- Avoid “boxing up” cells, usually 3 horizontal lines are enough: above, below, and after heading
- Avoid double horizontal lines
- Add enough space between rows

| Model | $C_L$ | $C_D$ | $C_{My}$ |
|---|---|---|---|
| Euler | 0.083 | 0.021 | -0.110 |
| Navier–Stokes | 0.078 | 0.023 | -0.101 |

Table 4.1: Table caption.

Make reference to Table 4.1.

Tables 4.2 and 4.3 are examples of tables with merging columns:

An example with merging rows can be seen in Tab.4.4.

If the table has too many columns, it can be scaled to fit the text widht, as in Tab.4.5.

| | Virtual memory [MB] | |
|---|---|---|
| | Euler | Navier–Stokes |
| Wing only | 1,000 | 2,000 |
| Aircraft | 5,000 | 10,000 |
| (ratio) | $5.0\times$ | $5.0\times$ |

Table 4.2: Memory usage comparison (in MB).

| | $w = 2$ | | | $w = 4$ | | |
|---|---|---|---|---|---|---|
| | $t = 0$ | $t = 1$ | $t = 2$ | $t = 0$ | $t = 1$ | $t = 2$ |
| $dir = 1$ | | | | | | |
| $c$ | 0.07 | 0.16 | 0.29 | 0.36 | 0.71 | 3.18 |
| $c$ | -0.86 | 50.04 | 5.93 | -9.07 | 29.09 | 46.21 |
| $c$ | 14.27 | -50.96 | -14.27 | 12.22 | -63.54 | -381.09 |
| $dir = 0$ | | | | | | |
| $c$ | 0.03 | 1.24 | 0.21 | 0.35 | -0.27 | 2.14 |
| $c$ | -17.90 | -37.11 | 8.85 | -30.73 | -9.59 | -3.00 |
| $c$ | 105.55 | 23.11 | -94.73 | 100.24 | 41.27 | -25.73 |

Table 4.3: Another table caption.

| ABC | header | | | |
|---|---|---|---|---|
| | 1.1 | 2.2 | 3.3 | 4.4 |
| IJK | group | | 0.5 | 0.6 |
| | | | 0.7 | 1.2 |

Table 4.4: Yet another table caption.

| Variable | a | b | c | d | e | f | g | h | i | j |
|---|---|---|---|---|---|---|---|---|---|---|
| Test 1 | 10,000 | 20,000 | 30,000 | 40,000 | 50,000 | 60,000 | 70,000 | 80,000 | 90,000 | 100,000 |
| Test 2 | 20,000 | 40,000 | 60,000 | 80,000 | 100,000 | 120,000 | 140,000 | 160,000 | 180,000 | 200,000 |

Table 4.5: Very wide table.

### 4.3.4 Mixing

If necessary, a figure and a table can be put side-by-side as in Fig.4.4

| Legend | | |
|---|---|---|
| A | B | C |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Figure 4.4: Figure and table side-by-side.

# Chapter 5

# Conclusions

Insert your chapter material here...

## 5.1 Achievements

The major achievements of the present work...

## 5.2 Future Work

A few ideas for future work...

# Bibliography

[1] IoT: number of connected devices worldwide 2012-2025. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. (Accessed on 11/18/2018).

[2] SMALT - The Smart Home Device That Elevates Your Dining Experience. Electronic Gadget, 2017. URL http://www.mysmalt.com/. (Accessed on 01/05/2018).

[3] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, et al. Understanding the mirai botnet. 2017.

[4] B. Krebs. Reaper: Calm before the iot security storm? — krebs on security. https://krebsonsecurity.com/2017/10/reaper-calm-before-the-iot-security-storm/, 10 2017. (Accessed on 01/04/2018).

[5] Addition to tls 1.3 contributors list. https://github.com/tlswg/tls13-spec/commit/43461876882a60251ecf24fb097f0ce2d7be4745. (Accessed on 01/11/2018).

[6] E. Rescorla. The transport layer security (tls) protocol version 1.3. RFC 8446, RFC Editor, August 2018.

[7] Commits by iluxonchik tlswg/dtls13-spec. https://github.com/tlswg/dtls13-spec/commits?author=iluxonchik. (Accessed on 11/18/2018).

[8] Nvd - cve-2018-1000520. https://nvd.nist.gov/vuln/detail/CVE-2018-1000520. (Accessed on 10/10/2018).

[9] ssl_server.c and ssl_client1.c are using an sha-1 signed certificate. https://github.com/ARMmbed/mbedtls/issues/1519. (Accessed on 10/15/2018).

[10] update test rsa certificate to use sha-256 instead of sha-1 by iluxonchik. https://github.com/ARMmbed/mbedtls/pull/1520. (Accessed on 10/15/2018).

[11] S. Raza, L. Seitz, D. Sitenkov, and G. Selander. S3k: Scalable security with symmetric keys—dtls key establishment for the internet of things. *IEEE Transactions on Automation Science and Engineering*, 13(3):1270–1280, 2016.

[12] C. Bormann, S. Lemay, H. Tschofenig, K. Hartke, B. Silverajan, and B. Raymor. CoAP (Constrained Application Protocol) over TCP, TLS, and WebSockets. Internet-Draft draft-ietf-core-coap-tcp-tls-11, IETF Secretariat, December 2017. URL `http://www.ietf.org/internet-drafts/draft-ietf-core-coap-tcp-tls-11.txt`. `http://www.ietf.org/internet-drafts/draft-ietf-core-coap-tcp-tls-11.txt`.

[13] mbed TLS (formely PolarSSL): SSL/TLS Library For Embedded Devices. URL `https://tls.mbed.org/`. (Accessed on 12/19/2017).

[14] Qualys ssl labs - ssl pulse. `https://www.ssllabs.com/ssl-pulse/`. (Accessed on 09/08/2018).

# Appendix A

# Vector calculus

In case an appendix if deemed necessary, the document cannot exceed a total of 100 pages...

Some definitions and vector identities are listed in the section below.

## A.1 Vector identities

$$\nabla \times (\nabla \phi) = 0 \tag{A.1}$$

$$\nabla \cdot (\nabla \times \mathbf{u}) = 0 \tag{A.2}$$

# Appendix B

# Technical Datasheets

It is possible to add PDF files to the document, such as technical sheets of some equipment used in the work.

## B.1 Some Datasheet

SUNPOWER

# C60 SOLAR CELL

MONO CRYSTALLINE SILICON

## BENEFITS

### Maximum Light Capture

SunPower's all-back contact cell design moves gridlines to the back of the cell, leaving the entire front surface exposed to sunlight, enabling up to 10% more sunlight capture than conventional cells.

### Superior Temperature Performance

Due to lower temperature coefficients and lower normal cell operating temperatures, our cells generate more energy at higher temperatures compared to standard c-Si solar cells.

### No Light-Induced Degradation

SunPower n-type solar cells don't lose 3% of their initial power once exposed to sunlight as they are not subject to light-induced degradation like conventional p-type c-Si cells.

### Broad Spectral Response

SunPower cells capture more light from the blue and infrared parts of the spectrum, enabling higher performance in overcast and low-light conditions.

### Broad Range Of Application

SunPower cells provide reliable performance in a broad range of applications for years to come.

**The SunPower™ C60 solar cell with proprietary Maxeon™ cell technology delivers today's highest efficiency and performance.** The anti-reflective coating and the reduced voltage-temperature coefficients provide outstanding energy delivery per peak power watt. Our innovative all-back contact design moves gridlines to the back of the cell, which not only generates more power, but also presents a more attractive cell design compared to conventional cells.

### SunPower's High Efficiency Advantage

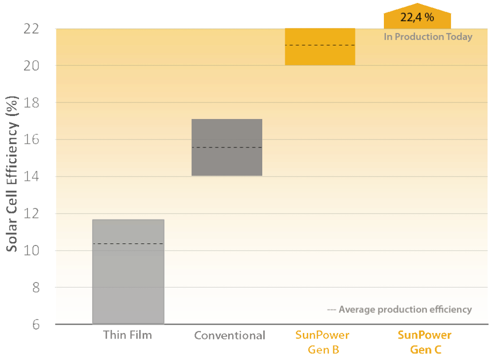
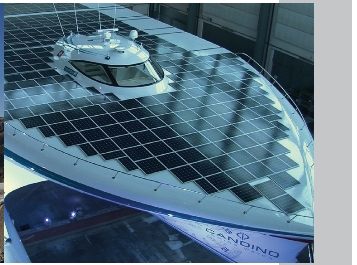


Photo courtesy of 3S Photovoltaics



C60 SOLAR CELL

SUNPOWER

C60 SOLAR CELL

MONO CRYSTALLINE SILICON

### Electrical Characteristics of Typical Cell at Standard Test Conditions (STC)

STC: 1000W/m², AM 1.5g and cell temp 25°C

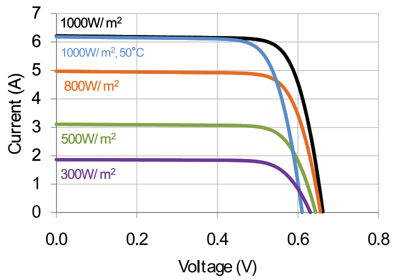
| Bin | Pmpp (Wp) | Eff. (%) | Vmpp (V) | Impp (A) | Voc (V) | Isc (A) |
|---|---|---|---|---|---|---|
| G | 3.34 | 21.8 | 0.574 | 5.83 | 0.682 | 6.24 |
| H | 3.38 | 22.1 | 0.577 | 5.87 | 0.684 | 6.26 |
| I | 3.40 | 22.3 | 0.581 | 5.90 | 0.686 | 6.27 |
| J | 3.42 | 22.5 | 0.582 | 5.93 | 0.687 | 6.28 |

All Electrical Characteristics parameters are nominal
Unlaminated Cell Temperature Coefficients
Voltage: -1.8 mV / °C Power: -0.32% / °C

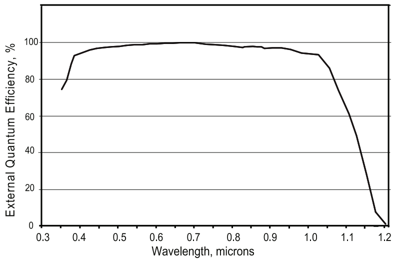
### Positive Electrical Ground

Modules and systems produced using these cells must be configured as "positive ground systems".

### TYPICAL I-V CURVE



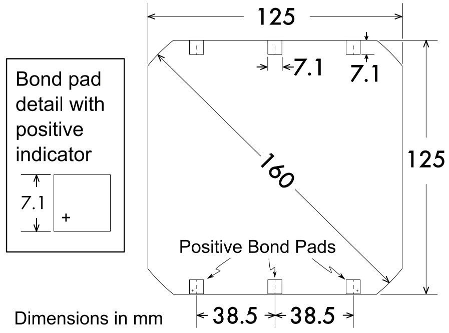
### SPECTRAL RESPONSE



### Physical Characteristics

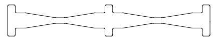
| | |
|---|---|
| Construction: | All back contact |
| Dimensions: | 125mm x 125mm (nominal) |
| Thickness: | 165μm ± 40μm |
| Diameter: | 160mm (nominal) |

### Cell and Bond Pad Dimensions



Bond pad area dimensions are 7.1mm x 7.1mm
Positive pole bond pad side has "+" indicator on leftmost and rightmost bond pads.

### Interconnect Tab and Process Recommendations



Tin plated copper interconnect. Compatible with lead free process.

### Packaging

Cells are packed in boxes of 1,200 each; grouped in shrink-wrapped stacks of 150 with interleaving. Twelve boxes are packed in a water-resistant "Master Carton" containing 14,400 cells suitable for air transport.

Interconnect tabs are packaged in boxes of 1,200 each.

### About SunPower

SunPower designs, manufactures, and delivers high-performance solar electric technology worldwide. Our high-efficiency solar cells generate up to 50 percent more power than conventional solar cells. Our high-performance solar panels, roof tiles, and trackers deliver significantly more energy than competing systems.

sunpowercorp.com