Reverse engineering of embedded consumer electronic systems

Ian McLoughlin

Abstract— **Cutting edge consumer electronics systems typically rely upon embedded processors and software for a large part of their competitive features. Developing such features is, in turn, a costly part of the design exercise that is usually recouped through initial high-price sales.**

If competitors are able to reduce their own development costs, and cut time-to-market through the reverse engineering of the pioneering products, such practices will distort the market and stifle forward progress. Unfortunately, reverse engineering of consumer products for nefarious purposes appears to be commonplace, with significant cost implications on industry sales and profitability.

This paper discusses the scope of the reverse engineering problem for consumer electronics reliant upon embedded processors, formalises the process of reverse engineering, and classifies potential mitigation strategies.

I. Introduction

New consumer electronics products only reach market after a long, arduous and potentially expensive design process. Companies releasing cutting-edge products tend to recoup development costs during the first few months of sales in an uncluttered market. In general, if other companies delay releasing competing devices, then profitibility is enhanced.

Of course, any pioneering design and manufacturing company (DMC) will naturally expect competing products to appear in time (and these may improves upon the original design in some ways). Often their competitors development costs will be as large as, if not greater than, their own. Assuming that manufacturing costs are similar for both devices, a profitable sales price will also be similar for both products.

However these market economics are significantly disrupted when a competitor cheaply and quickly reverse engineers a pioneering design. In this case, the competitors development costs are largely determined by the reverse engineering (RE) costs. If these RE costs are low, then the new product may easily undercut the pioneer device in price. This shortens the market lead of the pioneering product, but also massively curtails sales due to reduced market share (which is in turn a consequence of price undercutting). The assumption that RE costs are lower than development costs (i.e. the process is shorter, and less expensive than a full prototype-development project) is borne out in the evidence of commercial examples of design piracy and the prevalence of industrial espionage.

In general, the larger the amount of upfront development cost that can be saved through RE, the greater the risk to a pioneering company [1]. It is thus beneficial to analyse that difference by firstly understanding the RE process, and then applying this to a particular consumer electronic system under consideration.

Note that RE is not always done for nefarious purpose: many engineers, including the author, enjoy the reverse engineering process. In fact RE to understand, and improve upon, a competitors product is legal in many territories - as long as patents and copyrights are not infringed [2].

This paper is more concerned with the use of the RE process to commit design piracy, a negative outcome. Apart from being illegal, this stifles product innovation, and hinders forward progress. Although it is an attack upon the intellectual property of the DMC, it also affects the consumer as well as the wider consumer electronics industry.

The primary engineering response to the threat of design piracy through reverse engineering, is to incorporate RE protection (REP) into new products. The aim of REP is to increase RE cost as much as possible, with (hopefully) only a moderate increased development cost as a consequence. REP requires an appreciation of RE and the RE process, an identification of risks, and an output which is an engineering action to mitigate against these risks. This paper attempts to formalise the nomenclature of RE, with the follow on step of classifying REP approaches and eventual aim of allowing researchers to evaluate the effectiveness of REP for different stages of RE. Section II will discuss and define reverse engineering (RE) applied to consumer electronics in general, while section III provides a step-by-step overview of the RE process. Section IV introduces a simple economic analysis model, which is used in V to classify some of the more obvious mitigation techniques. Finally, section VI concludes the paper.

II. Reverse engineering defined

The RE process involves determining the functionality, architecture and technology of a system, represented in such a way that allows reuse or duplication of the original product, its architecture or technology. For consumer electronic systems, the end goal may be for many reasons, include the following:

- Understanding what the system does
- Understanding how the system works
- Replicating/copying/cloning part or all of the system
- To enable modification of the functionality of the system
- To obtain trade secrets
- To inform and educate
- For validation and/or verification

As mentioned in Section I, a common motivation for performing RE is economic, since the information gained can be used to significantly shorten the development cycle and time-to-market of a competing product. It is this motivation which is likely to have the greatest impact on the consumer electronic systems manufacturer, and it is thus RE performed for this purpose that we would most like to mitigate. Much of this discussion does not apply to RE for national

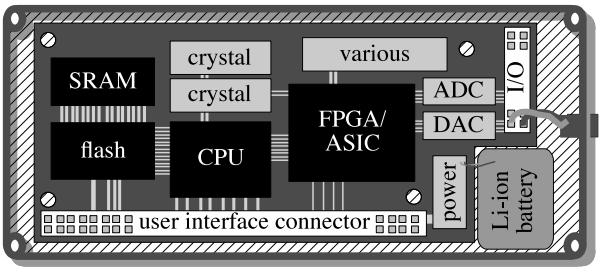

Fig. 1. Block diagram of a typical consumer electronics device containing an embedded processor

security purposes, since in such cases, there may be no economic argument for performing the RE, and thus no economical protection against it.

For other cases, especially commercially-motivated RE, we first state that it is impossible to build a completely secure system [3]. However it is very possible to construct a consumer electronic system that has sufficient REP incorporated in it to make the RE process uneconomical. The aim is that those who attempt to RE such systems would quite quickly recognise the futility of attempting the process and give up half way.

Hardening consumer electronics systems against RE may require increased development time and effort, increased manufacturing costs (including potentially a more expensive bill of materials), and often result in reduced serviceability. Engineers need to be able to understand the big picture behind these issues, and thus to understand the process of RE itself, as described in the following section.

III. The Reverse Engineering Procedure

Consumer electroncis RE involves several analytical steps which are not necessarily sequential, and depend to a large extent upon the components within the system being analysed. Since consumer electronics devices cover a very wide scope, we note firstly that we are predominantly interested in the RE of systems containing an embedded processor. The reason for this is that in such systems, a significant proportion of the functionality that sets the product apart from the competition, will often be implemented in software. To aid discussion, Fig. 1 presents a typical consumer electronic device consisting of a CPU connected to volatile memory (SRAM in this case), non-volatile memory (flash), a field-programmable gate array (FPGA) or application specific integrated circuit (ASIC), a user interface of some kind and a power unit. Of course not all systems will contain all of these, but this paper presents this example system as an aid to further discussion.

The top-down RE of a consumer electronics device such as that shown in Fig. 1 may include determining the information associated with the following categories:

- **A Functionality** - understanding what the system does
- **B Physical structure analysis**:
  - B.1 electro-mechanical arrangement
  - B.2 enclosure design
  - B.3 printed circuit board layout
  - B.4 wiring looms and connectors
  - B.5 assembly instructions
- **C Bill of materials**:
  - C.1 active electronic components
  - C.2 passive electronic components
  - C.3 interconnect wires and connectors
  - C.4 mechanical items
- **D System architecture**:
  - D.1 functional blocks and their interfaces
  - D.2 connectivity
- **E Detailed physical layout**:
  - E.1 placement of individual components
  - E.2 electrical connectivity between components
  - E.3 impedance controlled and location-aware orientation
- **F Schematic of electrical connectivity** - system circuit diagram.
- **G Object/executable code** - including:
  - G.1 isolation of code processors
  - G.2 isolation of firmware code for reconfigurable logic
- **H Software** - analysis of the software within the object files, including embedded code, bootloader, firmware plus ASIC reverse engineering.

Most of the item categories listed here are self-explanatory, and several others are described either in [1] (or in [4], chapter 7). The final category, **H**:Software RE, in general is a very well researched field. Chikofsky and Cross [5] provide an excellent overview, and taxonomy, and thus this present paper does not attempt a detailed taxonomy for software in consumer electronic systems, however there are differences between general purpose software, and that developed for personal computers. In the first place, consumer electronics software is usually resident in flash memory (as shown in Fig. 1), rather than on hard disc. The operating systems are also vastly different. Most desktop personal computers run Microsoft Windows, which is an inherently insecure OS, and one that is often exposed to almost unlimited external connectivity to the Internet. However most large scale computers (i.e. server farms) run Linux, as do many consumer electronic devices such as televisions, entertainment devices, smartphones and so on. Linux, based upon a UNIX model, is a far more secure OS. Smaller and cheaper consumer electronics devices may run a traditional real-time operating system (RTOS) such as VxWorks, QNX, eCOS, ThreadX or similar. Traditional RTOS code usually exists as a single composite image that contains all functionality along with a kernel, and may be quite difficult to RE, exacerbated by the lack of high level analysis tools. Embedded Linux, by contrast, often involves a stand-alone kernel image, a filesystem, and a separate bootloader. These can be examined and analysed separately [4], which means that in the absence of *any* software security choices, the code may be vulnerable to analysis.

It may be that, at the benign end of the scale, software RE is a useful means to achieve the potential reuse of OO code, whereas at the opposite extreme, circumvention of copy protection schemes leads to software piracy and theft. Interestingly, studies have even included the question of the morality of software piracy, and attitudes to intellectual property violations, and related this to region [6]. There is no indication that the conclusions of such a study are confined to software only. It is interesting to note the anecdotal evidence that pirated consumer electronic devices are found more often in some regions of the world than in others.

Software plays an important role in embedded systems, and although it is advisable for manufacturers to consider software RE and software security, this will not be considered further here since, as has been mentioned, it is a subset of general software RE and protection [7]–[9].

IV. Economic analysis

From the point-of-view of a design and manufacturing company (DMC), any newly released product is at risk of design theft through RE by unscrupulous competitors. When such an attack occurs, the outcome will probably be that DMC revenue is reduced. One of the contributions of this paper is to demonstrate that the DMC can spend more money up-front, and possibly more on BOM costs, to reduce the likelihood of an RE-based attack and - if one does occur - to reduce the impact on their revenue.

In order to examine this effect, let E_{PD} be the non-recurring expenditure on product development by a DMC. C_M is the cost of manufacturing one unit of product, C_S is the per-unit revenue, with sales amounting to N units. From this, and ignoring all other

unrelated factors, DMC profit over a product lifetime, P_{DMC} is given in Eqn. 1

$$P_{DMC} = N(C_S - C_M) - E_{PD} \quad (1)$$

However a more realistic model would be for a manufacturer to change prices during a product lifetime as a consequence of a competitor product reaching market. Thus we will define C_{S1} as the sales cost over N_1 units from market introduction, reduced to C_{S2} over N_2 units once the first competition arrives, as in Eqn. 2:

$$P_{DMC} = N_1(C_{S1} - C_M) + N_2(C_{S2} - C_M) - E_{PD} \quad (2)$$

Next we assume that the first competitor released their product precisely because they cloned the DMC through RE. There may well be other competition joining the market later, but the very fact that RE can shorten a development cycle supports the assumption that, if cloning occurs due to RE, the worst case impact on the DMC will be the early introduction of lower cost competition. In reality the competition may not be a complete clone, but rather have copied aspects of the original design, however the rationale still stands: the use of RE means that the competition could thus be earlier and cheaper than a genuine in-house development.

The reverse engineering company (REC) cloning the product would experience similar C_S and C_M but replace the development cost E_{PD} with the non-recurring RE cost E_{RE} as in Eqn. 3 to yield a profit P_{REC}, assuming a 50% market share:

$$P_{REC} = N_2(C_{S2} - C_M) - E_{RE} \quad (3)$$

Quite clearly, the profit made by the DMC over and above the REC is related to the difference between development and RE cost from Eqns. 2 and 3:

$$\begin{aligned} P_{DIFF} = & \quad P_{DMC} - P_{REC} \\ = & \quad N_1(C_{S1} - C_M) - E_{PD} + E_{RE} \end{aligned} \quad (4)$$

It would then seem that any DMC performing up-front development of a pioneering consumer electronics device needs to maintain a sales monopoly as long as possible, and ensure that their product is difficult/time consuming/expensive to reverse engineer.

As mentioned in Section II, complete RE protection, requiring an unconditionally secure system, is impossible to achieve in practice, however it is very possible to increase E_{RE} by making the RE process more difficult. To do this the DMC will require extra design effort, and thus increased up-front E_{RP}, and possibly extra componentry or manufacturing complexity in each device sold, C_{RP}.

If the DMC chooses to adopt this approach of incorporating REP, their profit will now be as shown in Eqn. 5, where extra terms reducing this profit are included. However the entry to market of the competition is delayed, and so the sales quantities differ from Eqn. 2: N_1' and N_2', where $N_1' > N_1$ and $N_2' > N_2$ (the former because the REC enters the market later, and the latter because the REC costs are higher, and hence their sales price).

$$\begin{aligned} P'_{DMC} = N_1'(C_{S1} - C_M - C_{RP}) \\ +N_2'(C_{S2} - C_M - C_{RP}) - E_{PD} - E_{RP} \end{aligned} \quad (5)$$

However, if these measures have been succesful, the RE cost to the competitor has increased also. Their adjusted profit is shown in Eqn. 6 (assuming that they are sufficiently intelligent to skip the components included specifically for RE protection when they clone the device).

$$P'_{REC} = N_2'(C_{S2} - C_M) - E'_{RE} \quad (6)$$

The reduction in profit to the REC incured due to the RE protection, is thus equivalent to the difference in Eqn. 7.

$$\begin{aligned} P_{REC} - P'_{REC} &= N_2(C_{S2} - C_M) - E_{RE} \\ &\quad -N_2'(C_{S2} - C_M) + E'_{RE} \\ &= (N_2 - N_2')(C_{S2} - C_M) - (E_{RE} - E'_{RE}) \end{aligned} \quad (7)$$

$P_{REC} - P'_{REC}$ is largely based on two factors, namely the difference in sales units due to later market entry $\Delta N_2 = N_2 - N_2'$ and the increase in RE cost $\Delta E_{RE} = E_{RE} - E'_{RE}$, thus:

$$P_{REC} - P'_{REC} = \Delta N_2(C_{S2} - C_M) - \Delta E_{RE} \quad (8)$$

The increased cost to the DMC, and reduced profit is given in Eqn. 9 using similar notation:

$$\begin{aligned} P_{DMC} - P'_{DMC} = \\ = \Delta N_1(C_{S1} - C_M) \\ +\Delta N_2(C_{S2} - C_M) \\ +(N_1' + N_2')C_{RP} - E_{RP} \end{aligned} \quad (9)$$

Clearly the major factors increasing the change in profit are the cost of implementing the REP, both in terms of NRE and in ongoing expenses, offset by the change in sales volumes. In fact, it is possible to define a metric which identifies the effectiveness of REP, the ratio γ, between the decrease in profits of the two entities as given in Eqn. 10.

$$\begin{aligned} \gamma &= \frac{P_{DMC} - P'_{DMC}}{P_{REC} - P'_{REC}} \\ \gamma &= \frac{\Delta N_1(C_{S1} - C_M)}{\Delta N_2(C_{S2} - C_M) - \Delta E_{RE}} \\ &+ \frac{\Delta N_2(C_{S2} - C_M)}{\Delta N_2(C_{S2} - C_M) - \Delta E_{RE}} \\ &+ \frac{(N_1' + N_2')C_{RP} - E_{RP}}{\Delta N_2(C_{S2} - C_M) - \Delta E_{RE}} \end{aligned} \quad (10)$$

The measure of success of deliberate REP is thus a combination of several interlinked factors indended to minimise the ratio γ, namely:

- increasing the positive difference ΔN_2 and decreasing the negative difference ΔN_1 by making RE more difficult, such as to delay entry to the second phase of reduced profit caused by direct competition.
- limit any increases in E_{RP} and C_{RP} as much as possible consistent with increasing the incremental cost of reverse engineering, ΔE_{RE}.

It is unfortunate that the engineering answer to RE risk in consumer electronic systems, namely to incorporate REP, leads to lower profits all around. Socially speaking this is a lose-lose situation, except for the particularly welcome consequence of increasing the gainful employment of development engineers.

V. RE MITIGATION TECHNIQUES

Since RE is not preventable, the issue becomes primarily economic, as outlined in section IV: how can a DMC maximise the RE cost experienced by competitors at minimal additional cost to themselves. General system protection methods have been presented elsewhere [10], but here the variety of RE mitigation techniqes will be classified, alongside a rating of their difficulty of implementation. Most importantly, these are ranked based on their cost to a DMC, and the economic impact of their implementation upon a RE-based attacker.

TABLE I

PASSIVE METHODS OF INCREASING HARDWARE RE COST SCORED ON SEVERAL CRITERA, OUT OF 5, PLUS THE AREA OF INFORMATION THAT THEY PREDOMINANTLY INFLUENCE, FROM SECTION III.

Method	Design cost	RE cost	Manuf. impact	Area affected
Tamper proof fixtures	2	0	1	B
Potting	1	1	2	B
Remove silk screen	1	1	1	F
Remove component markings	1	1	2	C
Use BGA packages	1	3	3	E.2
Route on inner layers only	2	2	3	E.2
Random fill of PCB	2	2	0	E.2
Use blind/buried vias	2	2	4	E.2
Jumble buses	1	1	0	F
Route through ASIC	5	3	2	F
Route through FPGA	2	2	2	F
Remove debug port	1	1	2	H
Memory filling	2	2	0	G
Encrypt code	1	2	0	H
Use of custom logic	5	5	2	D.1
Add impedence control	4	4	2	E.3

Methods of REP can be classified into *passive* or *active* [1], which are those fixed at design time, and those able to change operation in response to attach, respectively. Most important, is the analysis of cost multipliers experienced by the REC, and how those relate to the various REP methods. In general, REP costs will rise as a result of:

- Increased labour cost due to greater time taken to RE the system
- Increased labour cost due to higher levels of expertise required
- Increased cost spent on the purchase or use of specialised equipment required for the RE

A. Passive RE mitigation

Table I lists several common passive methods of REP, also providing an indication of their upfront cost in design terms, the likely incremental RE cost that they will incur, the degree of manufacturing impact, and the major elements of the RE procedure that they influence (from Section III). The scoring used is a subjective estimate. When performing an audit for a particular consumer electronics devices, this should be adjusted based upon knowledge of the particulars of that device.

B. Active RE mitigation

Many of the passive methods of section V-A also have active variants: this means, for example, that memory filling in flash is replicated at runtime in SRAM, or that the particular arrangement of a jumbled bus changes during system operation. Electrical connectivity can be confused by using spare inputs and outputs from processors, FPGAs or ASICs to route signals which are not timing critical but which are funcionally critical. Jumbled address and data buses are a little difficult to reverse engineer, but dynamically jumbled buses provide a significant further level of complication. They add cost to incorporate active devices for the jumbling/de-jumbling.

ASICs are probably the ultimate device for REP, but even the humble FPGA can be relatively effective. There are many possibilities for using such devices in active REP, including for performing encryption, however even without this, soft cores which are completely custom [11], and perhaps company confidential, add a layer of security through the hidden instruction set architecture and code arrangement. For an even more extreme protection, the instruction set could be deliberately changed in every implementation among several product versions to prevent repeated RE of core program.

TABLE II

RELATIVE STRENGTH OF ACTIVE PROTECTION METHODS, OUT OF 5 WHEN IMPLEMENTED DYNAMICALLY OR STATICALLY, AND THE AREA IMPACTED.

Method	Fixed timing	Dynamic timing	Area
Information hiding	0	2	G
Routing through logic	3	5	F
Obfuscation	1	3	D
Deliberate confusion	3	4	H
Impedance matching	4	5	E.3

C. Active RE mitigation classification

The basic forms of RE mitigation in consumer electronics can be classified into multiple dimensions. One useful classification, as we have seen, is into active or passive mitigation. A second division may be into temporal or spatial methods of achieving the protection.

Some mitigation classes are as follows:

- Temporal methods
- Spatial methods
- Active methods
- Passive methods
- Information hiding
- Obfuscation

Information hiding employs existing resources in ways that attempt to conceal information from attackers. Obfuscation, normally a passive method (such as swapping the names of labels and functions within code, or jumbling the PCB silk screen annotations) can also be active when changing system bus connectivity [12] [13], or device pin usage (for example multiplexing an interrupt input pin with a signal output function). This involves the use of existing, or spare resources, to complicate the RE process by deliberately misdirecting the RE team.

Protection by confusion can also mean adding resources specifically to deliberately confuse attackers. For example, including large pseudo-random data transfers, out-of-order or redundant code reading [14] and so on.

Spatial methods operate at a placement or connectivity level, and include scrambling buses differently depending upon memory area, turning on or off signal path routing devices depending upon operating mode or clock phase.

Temporal methods confuse through altering the sequence and/or timing of events. For example, a boot loader that deliberately executes only a subset of fetched instructions (i.e. uses redundant instruction fetch). Another would be a memory management device able to prefetch code pages from memory and access these non-linearly. Some examples which combine these attributes are shown in table II.

Undoubtedly, active dynamic obfuscation methods are the most costly and perhaps the most effective to develop, debug, deploy and test. What is clear is that custom logic is a great resource for actively resisting RE attacks.

VI. CONCLUSION

This paper has discussed the RE of consumer electronic systems that contain embedded processors (i.e. the majority of systems). It has set out the reason why RE may take place, and classified the processes involved. An extensive economic analysis provides impetus for systems manufacturers, particularly of pioneering high-value systems with a large software component, to incorporate REP.

The paper then outlined several of the more common REP strategies, firstly static techniques, and secondly more complex dynamic methods, identifying which part of the RE process they impact, and classifying both the costs and degree of protection afforded.

Unfortunately, RE will probably always be a factor in electronic design, and can be applied for social good as well as for nefarious

purposes. As consumer electronics systems become more complex, and more reliant upon embedded software, the risks of RE, and the potential gains from nefarious practitioners, both rise. It is becoming more difficult for manufacturers to ignore these issues. In fact, consumer electronics companies struggling to succeed in a crowded marketplace should make an informed decision as to what degree and type of REP, if any, they choose to apply in their products. If they choose not to apply any protection, they need to appreciate the risks that they face in terms of design theft and loss of intellectual property.

This paper attempts to move some way towards informing that decision, and presents classifications of the stages of the RE process, along with a useful economic analysis, and a classification of REP methods, their costs, and effectiveness.

REFERENCES

[1] I. McLoughlin, "Secure embedded systems: The threat of reverse engineering," *Parallel and Distributed Systems, International Conference on*, vol. 0, pp. 729–736, 2008.

[2] T. J. Biggerstaff, "Design recovery for maintenance and reuse," in *Computer*, July 1989, pp. 36–49.

[3] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Information Theory*, vol. 22, pp. 644–654, Nov. 1976.

[4] I. V. McLoughlin, *Computer Architecture: an embedded approach*. McGraw-Hill, Jan. 2011.

[5] E. J. Chikofsky and J. H. Cross, "Reverse engineering and design recovery: A taxonomy," in *IEEE Software*, Jan. 1990, pp. 13–17.

[6] W. R. Swinyard, H. Rinne, and A. K. Kau, "The morality of software piracy: A cross-cultural analysis," in *Journal of Business Ethics*, vol. 9, no. 8, Aug. 1990, pp. 655–664.

[7] C. S. Collberg and C. Thomborson, "Watermarking, tamper-proofing, and obfuscation - tools for software protection," in *IEEE Trans. Software Engineering*, vol. 28, no. 8, Aug. 2002, pp. 735–746.

[8] G. Naumovich and N. Memon, "Preventing piracy, reverse engineering, and tampering," *Computer*, vol. 36, no. 7, pp. 64–71, 2003.

[9] D. Hwang, P. Schaumont, K. Tiri, and I. Verbauwhede, "Securing embedded systems," *IEEE Security & Privacy*, vol. 4, no. 2, pp. 40–49, 2006.

[10] J. Grand, "Practical secure hardware design for embedded systems," in *Embedded Systems Conference*, California, Apr. 2004.

[11] R. Shadich and I. V. McLoughlin, "A modular computational engine for communications processing," in *Australian Telecomms. and Networking Apps. Conference*, Melbourne, Australia, Dec. 2003.

[12] Y. Yu, J. Leiwo, and B. Premkumar, "Hiding circuit topology from unbounded reverse engineers," in *Lecture Notes in Computer Science*, 2006, pp. 171–182.

[13] E. Hannah, "Method and apparatus for conditionally obfuscating bus communications," USA Patent PCT/US2005/040 371, Nov. 4, 2005.

[14] X. Zhuang, T. Zhang, H.-H. Lee, and S. Pande, "Hardware assisted control flow obfuscation for embedded processors," in *Proc. 2004 Int. Conf. on Compilers, Architectures, Synthesis on Embedded Systems*, 2004, pp. 292–302.