WHITE PAPER

IHR WEG ZUR ROBUSTEN EMBEDDED-PLATTFORM

GINZINGER
electronic systems

INHALTSVERZEICHNIS

1.1 HERAUSFORDERUNGEN BEI EMBEDDED-SYSTEMS-PROJEKTEN

Die Komplexität von Embedded Systems steigt durch immer neue Anforderungen. Die Speerspitze bildet dabei das „Consumer-Umfeld": von Touch-Bedienung, stetig steigender Vernetzung mit Cloud-Anbindung bis hin zu leistungsfähigen Anwendungen mit kinderleichter Benutzbarkeit (Usability). Auch der Gesetzgeber schreibt laufend neue Standards und Regelwerke vor. Diese Anforderungen müssen bei der Entwicklung und über die gesamte Lebensdauer berücksichtigt werden. Zusätzlich erschwert derzeit der überhitzte Bauteilemarkt die Versorgungsseite.

Dieses Spannungsfeld lässt Embedded-Systems-Projekte immer herausfordernder werden. Wie man trotz aller widrigen Umstände eine erfolgreiche und robuste Embedded-Plattform entwickeln kann, soll dieses Whitepaper beschreiben

1 IHR WEG ZUR ROBUSTEN EMBEDDED-PLATTFORM

Die Komplexität von Embedded Systems steigt durch immer neue Anforderungen.

2.1 WAS IST EIN EMBEDDED SYSTEM?

2.1.1 DEFINITION EMBEDDED SYSTEM

Als eingebettete Systeme (Embedded Systems) versteht man Computer oder Recheneinheiten, die in eine technische Umgebung eingebunden sind. Dabei verbindet ein solches System die große Flexibilität von Software mit der Leistungsfähigkeit der angepassten Hardware. Eingebettete Systeme verrichten – weitestgehend unsichtbar für den Benutzer – ihren Dienst in einer Vielzahl von Anwendungsbereichen und Geräten, beispielsweise in Industrie, Maschinenbau, Avionik, Bahn, Schiff, Heimanwendungen, Unterhaltungsindustrie etc.[1]

2 ERKLÄRUNG & HINTERGRUND

[1] Quelle: https://de.wikipedia.org/wiki/Eingebettetes_System

GINZINGER electronic systems

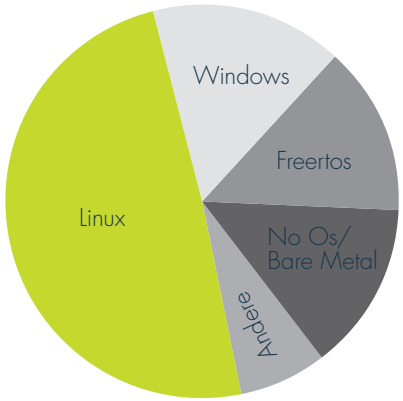
Linux-basierte Betriebssysteme dominieren aktuell den Bereich der „Internet of Things" (IoT)-Devices

2.2 WELCHE ROLLE SPIELT LINUX BEI EMBEDDED SYSTEMS?

2.2.1 LINUX ALS DE-FACTO-STANDARD IM EMBEDDED- UND IOT-SEKTOR

Der Embedded-Markt wird in den letzten Jahren immer stärker von Linux-basierten Betriebssystemen dominiert. Im jungen Bereich der „Internet of Things" (IoT)-Devices liegt man aktuell bei ca. 80 %.[1]
Grund für diesen Erfolg sind Performance, Verfügbarkeit und Support durch Fachfirmen und Community.

[1] https://www.itprotoday.com/iot/survey-shows-linux-top-operating-system-internet-things-devices



2.2.2 OPEN-SOURCE-LIZENZEN

Für viele ist das Thema Open-Source-Lizenz neben der Technik anfangs die größte Herausforderung in einem Embedded-Linux-Projekt. Droht doch bei Nichtbeachtung die Gefahr, seinen eigenen Code offenlegen zu müssen oder Rechtsverstöße zu begehen. Durch die Vielzahl an Open-Source-Lizenzen ist das Thema anfangs schwer zu fassen, mit dem richtigen Partner und unterstützendem Tooling jedoch handhabbar.

2.2.3 MUSS ICH FÜR LINUX BEZAHLEN?

Linux als Open-Source-Software steht grundsätzlich jedem frei zur Verfügung. Kostenpflichtig sind dabei – je nach Distribution und Ausprägung – die jeweilige Projektimplementierung, Stücklizenzen, Wartung und Security-Updates oder Supportdienstleistungen.

GINZINGER electronic systems

2.3 VERNETZTE GERÄTE UND SICHERHEIT

2.3.1 WELCHE TECHNIK ERFORDERT DIE IMMER WEITER UM SICH GREIFENDE VERNETZUNG?

Der steigende Grad an Vernetzung und daraus bedingte Sicherheitsfragen (Security) sind für so manche Projekte eine erhebliche Herausforderung. Vernetzung und Security gelten besonders bei „IoT" und „Industrie 4.0" als kritische Komponenten. Aber auch in klassischen Gebieten, wie im Maschinenbau, der Automatisierung oder bei Medizinprodukten, wird das Themengebiet rund um Vernetzung und Sicherheit immer prominenter.

Der steigende Grad der Vernetzung erfordert Sicherheitskonzepte.

3 HERAUSFORDERUNGEN

3.1 KOMPLEXITÄT

3.1.1 KOMPLEXITÄT DER EINGESETZTEN TECHNOLOGIEN

Die Handbücher von hoch integrierten elektronischen Bauteilen umfassen rasch mehrere tausend Seiten. Immer höhere Geschwindigkeiten zwischen Prozessor und Speicher erfordern diffizile Messmethoden. Eine neue Gehäusemechanik aus den unterschiedlichen Metallen und Kunststoffen oder edle Glasoberflächen für die Touch-Eingabe werden benötigt. Die Möglichkeiten in der Technik sind überwältigend, aber damit auch das Risiko, sich darin zu verlieren.

3.1.2 EFFIZIENTE SOFTWARE FÜR SCHLANKE HARDWARE

Will man bei Embedded Hardware kosteneffiziente Komponenten verwenden, muss man Kompromisse eingehen. Daher sind im Gegensatz zum Desktop Rechenleistung und Speicher oft nur eingeschränkt vorhanden. Um dennoch die ideale Benutzbarkeit zu bieten, müssen das Betriebssystem und die Hardwareplattform optimal aufeinander abgestimmt sein. Dieses optimale Gleichgewicht zu finden zeichnet gute Systemarchitekten aus. Denn nur durch die richtige Balance kann ein Produkt erfolgreich funktionieren und vermarktet werden.

Die Koordination und Abstimmung von unterschiedlichen Technologielieferanten ist eine große Herausforderung.

3.1.3 JE MEHR ANSPRECHPARTNER, DESTO KOMPLEXER

Teilt man die vielschichtigen Themen der Entwicklung auf unterschiedliche Lieferanten auf, wird man mit der Zeit erkennen, dass man zusätzliche Komplexität geschaffen hat. Die Koordination und Abstimmung zwischen den Technologielieferanten ist eine große Herausforderung, insbesondere im Falle eines Fehlers. Die Gefahr ist groß, dass jeder Beteiligte die Ursache beim anderen sieht, was Zeit kostet und die Lösung der Fehlerursachen behindert. Darum gilt es eine Projektstruktur zu finden, die eine reibungslose Umsetzung unterstützt.

3.1.4 INDUSTRIALISIERUNG

Als Industrialisierung versteht man die Abstimmung der entwickelten Hardware auf die Serienproduktion. Dazu gehören neben dem fertigungsgerechten Hardwaredesign auch das Test- und Softwareladekonzept. **Wichtig:** Nicht jeder Hardwareentwickler kennt die spezifischen Möglichkeiten einer Produktion. Eine enge Abstimmung mit dem EMS-Fertiger ist hier der Schlüssel zum Erfolg.

3.2 MARKTEINFÜHRUNG

3.2.1 ZEITPUNKT DER MARKTEINFÜHRUNG IST KRITISCH

Verpasst man einen wichtigen Messe- oder Kundentermin, „brennt der Hut“. Eine planmäßige Markteinführung ist essenziell für die erfolgreiche Vermarktung eines neuen Produktes. Der schnelle und einfache Start der Entwicklung hilft dabei ungemein. Viele Hersteller bieten hierfür Entwicklungs-Kits an. Dies reicht von universellen Plattformen wie Raspberry Pi, Beagle Board und Co. über spezifische Lösungen wie das Ginzinger-Entwicklungs-Kit auf Basis der i.MX-6-Familie. Neben dem richtigen Start sind natürlich auch die Umsetzung des Entwicklungsprojektes sowie die eingesetzten Technologien bestimmende Faktoren. Verwendet man eine geprüfte technische Basis mit guter Dokumentation, kann man über die gesamte Projektlaufzeit und Lebensdauer viel Zeit und Nerven sparen.

3.2.2 STAND DER TECHNIK – ERWARTUNGEN AUS DEM CONSUMER-UMFELD

Die Geschwindigkeit technischer Änderungen erfordert von Herstellern laufend Innovationen. Viele Kunden erwarten sich Features, die aus der Smartphone-Welt bekannt sind, wie zum Beispiel intuitive Bedienoberflächen, leistungsfähige Anwendungen und Cloud-Anbindung. Diese Features erfordern eine Vielzahl an technischen Grundlagen und eine komplexe Integration, um reibungslose Abläufe (User Experience) zu gewährleisten. Die vorhandenen Technologien clever zu verbinden kann den entscheidenden Vorteil auf dem Markt bringen.

3.3 VORSCHRIFTEN, NORMEN UND SICHERHEIT

3.3.1 WELCHE NORMEN TREFFEN FÜR DIE ENTWICKLUNG ZU?

Randthemen wie Vorschriften und Normen werden anfangs selten hinterfragt. Nur wenn alle diese Kostentreiber bereits zu Beginn identifiziert werden, ist gewährleistet, dass das Projekt später nicht unnötig ausufert. Gleich am Anfang ist zu klären:

- Welche Normen treffen für das Produkt, das geplante Einsatzgebiet und den Zielmarkt zu?
- Gibt es gesetzliche Rahmenbedingungen oder notwendige Zertifikate?
- Und sind Safety-Richtlinien einzuhalten, um einen sicheren Betrieb zu garantieren?

Welche Normen, gesetzliche Regelungen und Richtlinien sind zu beachten?

3.3.2 VERNETZUNG UND SECURITY

Die Vernetzung wird gemeinhin als herausforderndstes Thema bei aktuellen Embedded-Projekten gesehen.[1]
Vorreiter der Technologie sind neue Anwendungen wie Cloud-Dienste, Maschinenanalyse, Fernwartung oder Steuerung von Geräten via Smartphone und Tablet.

Bisher wurden viele Geräte isoliert vom Netzwerk betrieben, was sich zur Zeit rapide ändert, wodurch auch das Risiko von weitreichenden und teuren Security-Pannen steigt. Methoden zur Vermeidung solcher Pannen gibt es. Hierbei gilt es, je nach Risiko den richtigen Rahmen umzusetzen.

Achtung: Der Gesetzgeber plant, die Hersteller hier entsprechend in die Pflicht zu nehmen. Was Kalifornien schon vorgemacht hat[2], ist in Europa[3] bald zu erwarten. Daher muss Security von Anfang an mitberücksichtigt werden und müssen sichere Geräteupdates im Feld möglich sein.

[1] https://www.designnews.com/content/5-biggest-challenges-facing-embedded-software-developers-iot/144800967159182
[2] https://www.theverge.com/2018/9/28/17874768/california-iot-smart-device-cybersecurity-bill-sb-327-signed-law
[3] https://eur-lex.europa.eu/procedure/EN/2017_225

Besonders wichtig sind eine langfristige Unterstützung durch den Hersteller und die Auswahl der entsprechenden Komponenten.

3.4 LANGZEITUNTERSTÜTZUNG

3.4.1 UNTERSTÜTZUNG FÜR VIELE JAHRE – ANFORDERUNGEN AN HARD- UND SOFTWARE

Im industriellen Umfeld sind 10 bis 20 Jahre ein üblicher Produktlebenszyklus. „Consumer"-Komponenten mit typischen 2–3 Jahren Haltbarkeit sind hier kaum einsetzbar. Die längere Produktlebenszeit muss von Anfang an geplant werden:

- Stabile Software
- Update-Mechanismen für Anwendung und Betriebssystem
- Solide Hardwareplattform
- Einfache Möglichkeit des Debuggings
- Besonders wichtig: die langfristige Unterstützung durch die Hersteller und die Auswahl der richtigen Komponenten

3.4.2 VERFÜGBARKEIT UND ABKÜNDIGUNGEN VON BAUTEILEN

Bereits beim Gerätedesign muss darauf geachtet werden, dass verfügbare und langlebige Bauteile, möglichst aus mehreren Quellen, eingesetzt werden.

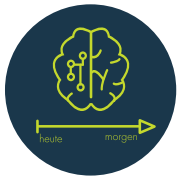
Bei einer Abkündigung muss rasch reagiert werden können, um die Lieferfähigkeit sicherzustellen. Im Idealfall – dank guter Komponentenauswahl und entsprechender Systemarchitektur – bemerkt der Endanwender diese Änderungen gar nicht.

3.4.3 ERWEITERUNGEN AN BESTEHENDEN PRODUKTEN

Einfache Erweiterbarkeit von Produkten ist neben Langzeitunterstützung der Schlüssel für geringe Kosten über den Lebenszyklus. Sind einige Jahre ins Feld gezogen, kann ein neues Release der Plattform ohne große Anpassung neue Funktionen bieten, den Grad der Vernetzung erweitern oder dem Kunden Kompatibilität mit neuen Standards bieten. Wenn die Architektur der Plattform stimmt, wird das idealerweise nur mit kleinen Softwareanpassungen realisierbar sein.

4 DIE DREI KERNFRAGEN

Bevor man für seine angestrebte Entwicklung den idealen Umsetzungsweg sucht, sollte man sich die folgenden drei Fragen stellen:



Welches Know-how ist vorhanden, welches kann aufgebaut werden?

Welches Know-how stiftet Kundennutzen?

Welche Ressourcen bleiben für die Entwicklung der Kernanwendung?

4.1 WELCHES KNOW-HOW BESITZEN WIR IM MOMENT, WELCHES WOLLEN WIR FÜR DIE ZUKUNFT AUFBAUEN?

- Welches Know-how ist bereits in der Entwicklung, im Einkauf und im Projektteam vorhanden?
- Wurde beispielsweise schon hardwarenahe Software entwickelt?
- Sind erfahrene Hardwareentwickler im Team?
- Ist der Einkauf darauf vorbereitet, in einem aufgeheizten Bauteilemarkt Komponenten zu besorgen und eine langfristige Lieferfähigkeit sicherzustellen?

Falls nicht, muss geklärt werden, ob man dieses Know-how im Haus aufbauen oder über externe Experten zukaufen will.

4.2 MIT WELCHEM KNOW-HOW STIFTEN WIR KUNDENNUTZEN UND VERDIENEN GELD BEI GERINGEM RISIKO?

Hier dreht sich alles um das Verhältnis Aufwand und Risiko in Relation zum Kundennutzen. Ist es sinnvoll, mühsam Wissen über Technologien für ein einmaliges Projekt aufzubauen? Kann man die Vielzahl an komplexen Entwicklungsherausforderungen abdecken? Oder ist es sinnvoller, Ressourcen beim direkten Kundennutzen zu bündeln? Dieser findet sich im Embedded-Projekt auf der Seite der Anwendung, in einer speziellen Sensorik oder in eigens entwickelten Verfahren und Algorithmen.

4.3 WIE VIEL RESSOURCEN BLEIBEN FÜR DIE ENTWICKLUNG DER KERNANWENDUNG?

Bleibt noch zu klären, ob überhaupt ausreichend Entwicklungsressourcen für dieses breite Feld an Aufgaben vorhanden sind. Oft wird anfangs der Aufwand für eine stabile Hardwareplattform und für eine entsprechende Softwarebasis unterschätzt: Zeit, die später in der kritischen Phase des Projektes fehlt.

5 LÖSUNGSWEGE

5.1 DIE DREI MÖGLICHKEITEN

Für die Umsetzung von Embedded-Systems-Projekten gibt es im Wesentlichen drei Lösungswege.

Komplette Eigenentwicklung der Hard- und Softwareplattform

Aufbau auf bestehendem Betriebssystem mit eigener oder fremder Hardware

Komplettanbieter kümmert sich um Hard- und Softwareplattform

5.2 KOMPLETTE EIGENENTWICKLUNG DER HARD- UND SOFTWAREPLATTFORM

„Selbst ist der Entwickler" lautet die Devise bei dieser Variante. „Selber machen" inkludiert die Entwicklung und Pflege der Hardware, der Treiber, des Betriebssystems sowie der Anwendung. Der große Vorteil ist, dass man das System am Ende in- und auswendig kennt und beherrscht. Startet man aber „von der untersten Ebene" weg, muss einem klar sein, dass man für alle kleinen und großen Anpassungen für alle Zeiten selbst verantwortlich ist. Das ist enorm zeitaufwendig und fehleranfällig. Bei einer Eigenentwicklung der Plattform muss Folgendes vor dem Projektstart geklärt sein:

- Ist Expertise in Hard- und Softwareentwicklung im Embedded-Umfeld bis runter zum Low-Level-Treiber verfügbar?
- Sind alle notwendigen Technologien bekannt und beherrschbar?
- Ist die Einkaufsabteilung fit für den Hardwareeinkauf?
- Hat die Hardwareentwicklung Erfahrung mit der serientauglichen Entwicklung einschließlich Testkonzepte und Fertigungsoptimierung?
- Ist ein Projektleiter verfügbar, der das Embedded-Projekt in Qualität, Zeit und Budget durchgehend koordinieren kann?
- Wie viel Zeit und Ressourcen will man gegebenenfalls für den Aufbau des oben genannten Wissens investieren?

Achtung bei Entwicklung mit temporären Kräften und Praktikanten: Sollten diese wichtige Teile der Plattform entwickeln, ist das Risiko groß, nach dem Austritt kritisches Wissen zu verlieren.

Eine planmäßige Markteinführung kann bei einer kompletten Eigenentwicklung mit genug Expertise gelingen. Es ist essenziell, die notwendigen Technologien nicht zu unterschätzen. Erkennt man Komplexität zu spät, kann die dadurch verlorene Zeit nicht mehr aufgeholt werden. Die Verantwortung für Normen und Regularien liegt komplett in der eigenen Hand. Kennt man das normative Umfeld noch nicht, lohnt es sich, vor dem Entwicklungsstart Rat vom Experten zu holen. Ist das Gerät vernetzt, muss selbst ein Security- und Update-Konzept erstellt und umgesetzt werden. Spätere Implementierungen sind schwierig und kommen teuer.

Life-Cycle-Pflege funktioniert, wenn genug Entwicklungsressourcen vorhanden sind, um neben dem täglichen Geschäft Zeit für Pflege, Weiterentwicklung und Wartung zu haben. Eine einfach erweiterbare Plattform muss von Beginn an als solche gestaltet sein. Das kostet anfänglich Zeit und Kosten und rechnet sich erst über die entsprechende Lebenszeit der Plattform.

Ob dieser Lösungsweg der richtige ist, hängt maßgeblich vom Grad des Know-hows im Unternehmen ab, aber auch davon, ob man dieses als Kernkompetenzen aufbauen und erhalten möchte. Eine vollständige eigene Plattformentwicklung für nur ein Produkt oder einen Produkttypus, ist kaum wirtschaftlich. Der breite Wissensaufbau hat es in sich. Gerade am Anfang läuft man Gefahr, viel „Lehrgeld" zahlen zu müssen und zeitliche Abschätzungen schwer einhalten zu können.

+ VORTEILE

- Umfassendes, lückenloses Systemverständnis
- Komplettes Know-how im Haus
- Alles aus einer Hand

– NACHTEILE

- Management und Verständnis der Vielzahl an komplexen Technologien
- Aufwand für Aufbau und Erhalt von Wissen
- Erst beim Einsatz in mehreren Produkten wirtschaftlich
- Alle Fehler sind selbst zu beseitigen
- Aufwand für Wartung und Pflege über die gesamte Lebensdauer

5.3 AUFBAU AUF BESTEHENDEM BETRIEBSSYSTEM MIT EIGENER ODER FREMDER HARDWARE

Alternativ zum kompletten Eigenbau besteht die Möglichkeit, bestehende kommerzielle oder freie Embedded-Betriebssysteme zu verwenden. Die Basis ist sofort verfügbar, die Hardware kann selbst oder außer Haus entwickelt werden. Da die Softwarepakete (Distributionen) selten auf das Endgerät abgestimmt sind, muss das Betriebssystem an die Hardware angepasst werden. Bei Linux kann man auf bekannte Konfiguratoren wie Yocto setzen oder auf von Chipherstellern voroptimierte Distributionen mit Board Support Packages (BSPs) setzen. Yocto bietet als System-Builder-Tool viele Möglichkeiten für die Anpassung des Betriebssystems auf die entsprechende Hardware, bringt aber im gleichen Maße unnötige Komplexität ins Spiel. Mit generischen Distributionen wie z.B. Ubuntu Core oder Balena gelingen die ersten Gehversuche vergleichsweise schnell – die Anpassung an die entsprechende Hardware ist aber weiterhin schwierig und erfordert tiefgehendes Wissen. Auf eine Zielhardware vorangepasste Distributionen wie Ginzinger Embedded Linux (GELin) bieten schlüsselfertig leistungsfähige, flexible Softwarepakete für ausgewählte Prozessorfamilien.

Wichtig bei der Auswahl des richtigen Betriebssystems sind:

- Ausführliche Dokumentation und Unterstützung durch Hersteller und Community
- Entwicklungs-Kits mit Evaluation Boards, um vor Fertigstellung der Hardware bereits mit der Entwicklung der Software beginnen zu können
- Verfügbarkeit umfassender Programme, Bibliotheken und moderner Entwicklungsframeworks
- Skalierbarkeit für Low-End- bis High-End-Anwendungen
- Langfristig verfügbare Software, Wartung und Support
- Einfache Hard- und Software-Integration
- Unterstützung von Frameworks wie z.B Qt für Bedienoberflächen

Der Aufbau auf bestehende Betriebssysteme nimmt viel Arbeit auf den untersten Ebenen der Softwareentwicklung ab und unterstützt eine schnelle Markteinführung, wenn die geplante Hardware kompatibel ist. Die technische Komplexität liegt hier in der robusten Abstimmung von Betriebssystem und Hardware.

Bei Normen und Regularien ist man weiterhin auf sein eigenes Wissen angewiesen. Basisfeatures für Security sind bei den meisten Distributionen implementiert, diese müssen je nach Projekt noch in ein eigenes Security-Konzept entsprechend der Bedrohungsstufe eingearbeitet werden. Langzeitunterstützung erfolgt in der Regel auf Basis von Long Term Support Releases (LTS) – diese bedeuten zwei bis drei Jahre an Unterstützung. Wartung und Pflege bleibt einem außerhalb dieses Zeitrahmens größtenteils selbst überlassen.

Kann man auf erfahrene Hardwareentwickler zurückgreifen oder Standardhardware verwenden, lohnt sich die Verwendung eines schlüsselfertigen Betriebssystems. Das Zusammenspiel von Hard- und Software darf nicht unterschätzt werden. Während des Projektes kann es zu vielerei technischen Herausforderungen kommen, die weiterhin tiefes Know-how im Hinblick auf Hard- und Software erfordern. Wichtige Termine kommen damit schnell ins „Wanken."

+ VORTEILE

- Schneller und einfacher Start
- Meist relativ breite Unterstützung an Standardhardware
- Community- oder Hersteller-Support

– NACHTEILE

- Spezialtreiber nicht vorhanden
- Langzeitunterstützung sehr stark vom Hersteller abhängig
- Mangelnder Support für spezifische Hardware
- Migration auf neue Plattform aufwendig
- Teilweise zu umfangreiche und komplexe Konfiguratioren
- Mäßig optimierte Softwarepakete

5.4 KOMPLETTANBIETER KÜMMERT SICH UM HARD- UND SOFTWAREPLATTFORM

Schließlich gibt es die Variante, einen externen Dienstleister als Komplettanbieter für die Entwicklung und Betreuung der Hard- und Softwareplattform zu beauftragen. Er kümmert sich um ein zuverlässiges Grundgerüst. Wichtig bei der Auswahl des richtigen Komplettanbieters sind:

- Erfahrung in der Entwicklung von Embedded-Lösungen
- Robuste, skalierbare und bewährte Basis vorhanden
- Technologisches Wissen und Erfahrung in:
  - Embedded Betriebssystemen
  - CPU und Speicheranbindung
  - Vernetzung und Security
  - HMI und Touch-Technologie
  - Realtime-Anwendungen
  - Wireless-Anbindung
  - Zulassung und Zertifikaten
- Langfristiger Support und Unterstützung über die gesamte Lebensdauer
- Entwicklungs-Kits für den schnellen Start bei der Applikationsentwicklung
- Produktionserfahrene Entwickler, idealerweise moderne EMS-Produktion im Haus
- Erfahrung in der Industrialisierung und in der Erstellung von Testkonzepten
- Erfahrung im Projektmanagement und Umsetzung vom Prototyp bis zur Serie

Mit dem richtigen Komplettanbieter baut die eigene Embedded-System-Entwicklung auf einer robusten und bestehenden Basisplattform auf. Dadurch spart man Zeit und reduziert die Komplexität. Gibt es zu der bestehenden Basisplattform auch ein Evaluations-Entwicklungsboard, kann parallel zur Hardwareentwicklung sofort mit der Applikationssoftware begonnen werden. Somit kann wertvolle Zeit in der Umsetzung gewonnen werden, was zu einem Vorsprung beim Marktstart führt. Anwendungsspezifische Normen und Regularien sind Komplettanbietern in der Regel bekannt. Langzeitunterstützung wird je nach gewünschter Produktlebensdauer gemeinsam definiert und umgesetzt. Die Beauftragung eines Komplettanbieter ist besonders sinnvoll, wenn man sich als Gerätehersteller nicht in Details von Hard- und Softwareintegration verlieren möchte. Hier kann man sich mit voller Energie auf die eigenen Stärken und auf den Kundennutzen fokussieren.

+ VORTEILE

- Expertenlösung aus einer Hand – Hardware, Software, Test und Fertigung
- Fokus bleibt auf Nutzen des Endkunden
- Effiziente Projektabwicklung und optimiertes Serienprodukt
- Geringe „Total Cost of Ownership" über die gesamte Lebensdauer

– NACHTEILE

- Komplettanbieter fehlt Domänen-Know-how der Anwendung
- Detailliertes Wissen über Embedded Systems außer Haus

GINZINGER electronic systems

Ginzinger electronic systems kann Sie bei allen drei Möglichkeiten mit erstklassigen Lösungen begeistern

SIE ENTWICKELN IHRE PLATTFORM KOMPLETT SELBST?

Als Embedded-Experte unterstützt Ginzinger electronic systems Sie dabei, ein für die Serienproduktion optimiertes, testbares und produzierbares Gerät zu entwickeln. Dieses kann in unserem Hightech-EMS-Maschinenpark (www.ginzinger.com/ems) optimal gefertigt werden.

SIE SUCHEN EINE LINUX-DISTRIBUTION UND WOLLEN KOMPETENTE UND VERLÄSSLICHE UNTERSTÜTZUNG?

Mit dem Ginzinger Embedded Linux (ginzinger.com/GELin) sind Sie bestens beraten. Drei Typen von Entwicklungs-Kits erlauben es Ihnen, sofort mit der Entwicklung loszulegen. Oder Sie buchen einfach einen Workshop und erfahren, wie GELin Ihre i.MX-basierte Hardware zur idealen Plattformlösung werden lässt.

EIN KOMPLETTANBIETER FÜR ALLE FÄLLE!

Als Komplettanbieter bietet Ginzinger electronic systems mit seinem Partnernetzwerk Know-how und erprobte Lösungen an. Je nach Bedarf kann das Produkt gleich fertig getestet, mit Software beladen, montiert und direkt in Ihr Lager gesendet werden. Hier werden Hardware und Software aufeinander abgestimmt und für ihren Einsatz optimiert. Das bringt eine zukunftssichere Plattform und geringe Stückkosten. Als Lösungspartner mit über 25 Jahren an Erfahrung stehen wir auch in Zukunft bei allen Anliegen an Ihrer Seite.

6
IHR WEGBEGLEITER ZUR ROBUSTEN EMBEDDED-PLATTFORM

JETZT DEN EXPERTEN KONTAKTIEREN:
+43 77 23 54 22 oder
office@ginzinger.com

GINZINGER
electronic systems

DOK022-MA-11/2018

GINZINGER
electronic systems

Gewerbegebiet Pirath 16 / 4952 Weng im Innkreis / T +43 77 23 54 22 / office@ginzinger.com / www.ginzinger.com